UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFIT/CI/NR 88-109	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) REDUCING SOFTWARE COSTS THROUGH STANDARDIZATION OF DESIGN AND CONFIGURATION MANAGEMENT		5. TYPE OF REPORT & PERIOD COVERED MS THESIS 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) WILLIAM J. FETECH		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS AFIT STUDENT AT: TRINITY UNIVERSITY		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS		12. REPORT DATE 1988 13. NUMBER OF PAGES 145
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) AFIT/NR Wright-Patterson AFB OH 45433-6583		15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

DISTRIBUTED UNLIMITED: APPROVED FOR PUBLIC RELEASE

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

SAME AS REPORT

18. SUPPLEMENTARY NOTES

Approved for Public Release: IAW AFR 190-1
LYNN E. WOLAVER 20 July 88
Dean for Research and Professional Development
Air Force Institute of Technology
Wright-Patterson AFB OH 45433-6583

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

ATTACHED

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REDUCING SOFTWARE COSTS THROUGH STANDARDIZATION OF DESIGN AND CONFIGURATION MANAGEMENT

ABSTRACT OF THESIS

Presented to the Faculty of Trinity University
in Partial Fulfillment of the Requirements

For the Degree of

Master of Science

By

William J. Fetech, B.S.

The Air Force requires a centralized and cost effective method for designing and implementing new data processing systems. Similarly, the implementation must also be applied to current data processing systems. The basic premise focuses on the fact that computer hardware prices have steadily decreased over the last fifteen years; however, the cost of software has become more expensive during this same period. With this point in mind, it does not make sense to use a general purpose computer to perform a variety of functions. This is especially true if the hardware is not suited to perform a certain function

easily. The software, the most costly part of a data processing system, should not be constrained by the hardware. Additionally, when the same function is performed by several separate organizations, it makes sense to have compatible hardware at these locations. Thus, the same software can be utilized at these locations. Consequently, the standardization of hardware to insure the transportability of software requires a thorough investigation. Standardization and transportability are explored in depth. The focus is on several Department of Defense (DOD) computer systems. The ideas and strategies expressed in this thesis can be applied to any automated function.

REDUCING SOFTWARE COSTS THROUGH STANDARDIZATION OF DESIGN AND CONFIGURATION MANAGEMENT

William J. Fetech

APPROVED BY THE THESIS COMMITTEE:

Chair

APPROVED BY THE CHAIR OF THE DEPARTMENT:

Department of Computing and Information Sciences

APPROVED BY THE DEAN:

Division of Sciences, Mathematics,and Engineering

May 1987

REDUCING SOFTWARE COSTS THROUGH STANDARDIZATION OF DESIGN AND CONFIGURATION MANAGEMENT

By

William J. Fetech, B.S.

THESIS

Presented to the Faculty of Trinity University
in Partial Fulfillment of the Requirements

For the Degree of

Master of Science

TRINITY UNIVERSITY
May 1987

TABLE OF CONTENTS

ACKNOWLEDGMENTS

The author wishes to express his appreciation to Dr. Clifford J. Trimble, Chairman of the Thesis Committee, Dr. Paul Myers, and Dr. Richard Cooper for their assistance in preparing this thesis. Their efforts and assistance have been invaluable in completing this thesis.

Last, but most certainly not least, the author wishes to extend love and sincere thanks to his wife, Debbie, for the help and support she has provided in writing this paper.

LIST OF ILLUSTRATIONS

LIST OF TABLES

The views expressed herein are those of the author and do not necessarily reflect those of the United States Air Force or of the Department of Defense.

CHAPTER I

INTRODUCTION

Focus

The major focus of this thesis is to demonstrate that software expenditures can be controlled through use of proper planning, centralized configuration management, hardware standardization, and a good management structure. A strategy is developed and presented to incorporate these techniques in current data processing systems. The transition of data processing systems to this strategy cannot occur overnight, but requires proper planning over a period of time. This is especially true if more than a few incompatible hardware configurations are involved. This is the case with two of the systems discussed in Chapter IV of this paper.

Usage

This thesis can be used for several purposes. First and foremost, it can be used to standardize hardware when developing new software or when standardizing current hardware and software. Second, Chapter V addresses

approaches for conversion of software to a new environment. This could be as simple as conversion from COBOL-68 to COBOL-74. On the other hand, this conversion could be as complex as moving the software from an IBM 360 to a UNIVAC 1180. In addition, the chapters on management structures, configuration management, and software testing can be read independently of the others. Finally, the DOD can use Chapter IV to begin standardizing the Electronic Warfare Integrated Reprogramming System and the various wargaming systems.

CHAPTER II

COSTS IN THE DATA PROCESSING INDUSTRY

Over the past fifteen years there has been a major change in expenditures, relative to hardware and software, in the computer industry. Referring to Figure 1, in 1955 software consumed less than twenty percent of the total data processing budget in the U.S. while hardware consumed the other eighty percent. Conversely, by 1985 hardware accounted for less than twenty percent and software accounted for over eighty percent of the data processing budget. During 1976, U.S. companies spent $35 billion for data processing products and services. By 1980, these expenditures reached $90 billion and $139 billion in 1985. With the data processing industry having over 500,000 computers installed by 1985, the industry now accounts for seven percent of the gross national product (GNP) of the U.S. This represents an increase from 155,000 computers and three percent of the GNP since 1976 [7, p. 202]. Looking at U.S. Air Force expenditures in 1972, $1.5 billion or four percent of the total Air Force budget was spent on software. On the other hand, the Air Force spent less than one-third of that or $500 million on hardware

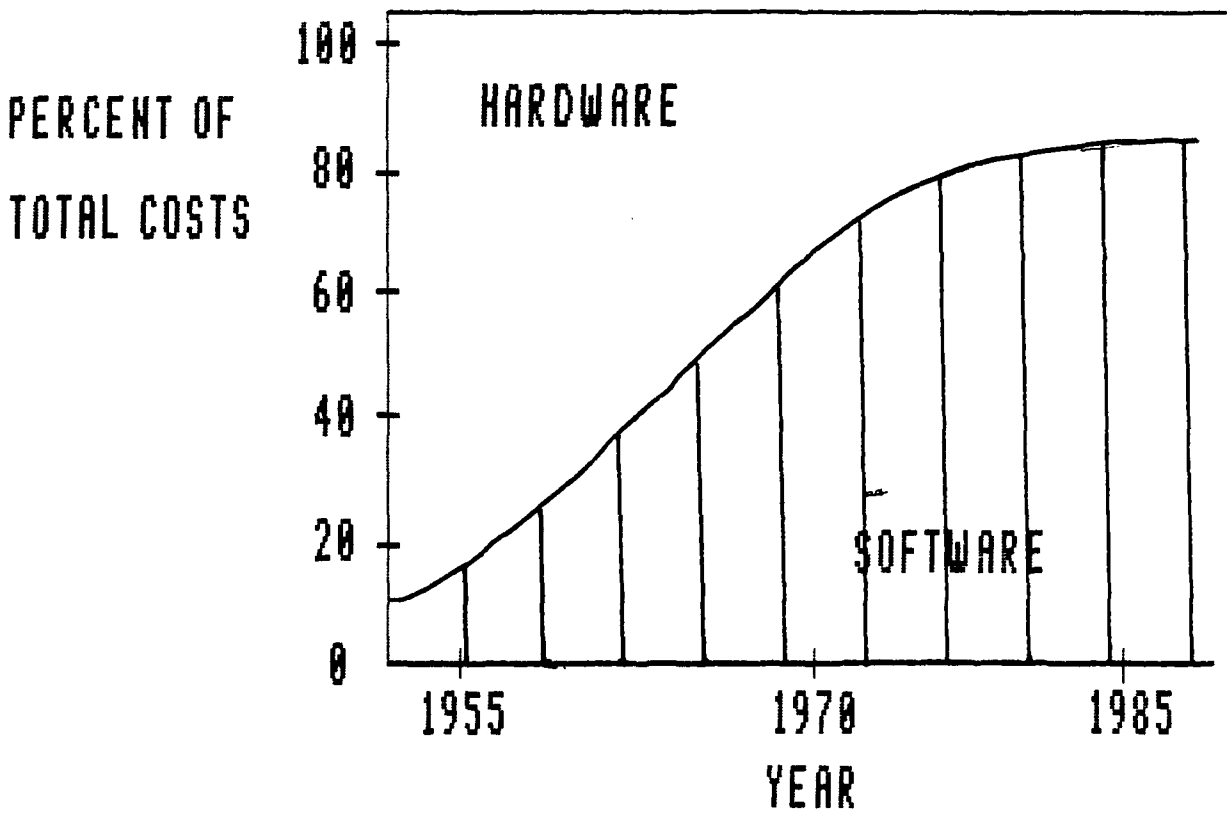

Figure 1. Hardware/software cost trends; relationships between hardware and software costs over the past thirty years [7, p.48].

[5, p. 48]. In summary, the expenditures in the data processing industry over the past fifteen years reveal that software costs have increased while hardware prices have decreased.

Hardware Costs [1]

In analyzing hardware costs, central processing units will be addressed first and then peripheral devices.

Central Processing Units

In 1983, a Burroughs B-7900 with twenty-four megabytes of memory sold for $2.5 million. By 1986, this same computer was only $1.7 million. In 1976, an IBM 370/138 with one megabyte of memory and 2 block multiplexer channels or an IBM 370/148 with 2 additional channels were purchased for $435,000 and $689,000 respectively. Today comparable systems having twice the memory may be procured for $39,600 and $49,600 respectively. The CDC CYBER Model 171 sold for $305,000 in 1976, but a comparable system, the Model 810A, now sells for only $121,000. The Univac 90/80 which rented for $47,000 per month in 1976 can be acquired for only $289,000. The rent in 1976 for that system over a seven month period pays for its replacement today. In

---

[1] The figures quoted on hardware prices were found in Datamation, Byte, and PC Magazine over the past fifteen years. Current prices were found by telephoning vendors.

1970, a Digital Equipment Corporation (DEC) PDP 11/10 and an 11/20 sold for $7,700 and $10,800 respectively. By 1980, the PDP 11/34, with thirty-two times more memory plus hardware multiply and divide capabilities was available for $9,290. In that same year, 1980, a PDP 11/70 could be purchased for $60,000. Today, the price for a micro PDP 11/73 is only $14,800.

This price reduction is not limited only to the large mainframe computers, but also to the micro and mini systems. In reviewing the microcomputer market, the price of an Apple III with ninety-eight kilobytes of memory, one floppy disk drive, and monitor was $4,400 in 1980. The Apple Lisa with a hard disk and two floppy disk drives sold for $9,995 in 1983. Today, the Apple III can be replaced by a system priced at $1,500. The Apple Macintosh XL will replace the Lisa for about $2,400. A Radio Shack TRS-80 Model II, 112, and 16 were priced at $2,999, $3,999, and $4,999 respectively in 1983. The price today for similar systems is $1,125 for both the Model II and 112 and $3,499 for the Model 16. Now examining the minicomputer market, the Data General MV/8000 sold for $153,150 in 1980 and the MV/10000 $325,000 in 1983. These systems now sell for $91,000 and $167,000 respectively. The price for a Wang 3300 has decreased from $17,500 in 1970 to only $12,000 today.

Summarizing the data presented, the large mainframe computers have decreased between thirty and ninety-two percent during the last ten years. Furthermore, the micro/mini computer prices have decreased between thirty and seventy-five percent in the last fifteen years.

Peripherals

In addition to the prices' of central processing units decreasing, the associated peripheral equipment has followed a similar trend.

Monitors

The DEC product line of monitors shows a steady decrease in price since 1980. In that year, a VT50 was purchased for $1,300, a VT52 $2,095, and a VT100 $1,895. By 1980 the VT100, which now replaced the VT50 and 52, was reduced to $1,695 and today may be purchased for only $895. The VT132 available in 1980 for $2,295 is replaced today for only $995. Other monitors with similar capabilities cost $2,750 in 1976, $1,675 to $1,799 by 1980, and $1,400 in 1983. For the VT line of terminals this equates to a decrease between thirty-one and fifty-seven percent since 1980. Since 1976, the VT compatible monitors have decreased sixty-seven percent. The VT type of monitors do have additional capabilities; consequently, the standard monitors will be reviewed separately. The

standard features of these monitors include an eighty character wide by twenty-five line screen, monochrome color, a keyboard, and a communications port. In 1970, the price of this type of monitor was between $3,500 and $5,000. By 1976, the price for a standard monitor decreased to $995 and to $749 by 1980. Today, a monitor having the same capabilities is priced about $325. This represents a decrease of ninety percent since 1970.

Printers

The price of a three hundred line per minute printer was $69,500 in 1970; however, today a six hundred line per minute printer brings between $12,900 and $15,400. Laser printers which were purchased for about $44,000 in 1976 can now be acquired for $16,000 to $28,000. Daisy wheel printers have been reduced from $1,700 in 1976 to around $420 in today's market. Prices for dot matrix printers have dropped from between $3,000 and $4,000 in 1976 to around $1,000 by 1983. By 1986, an average price for a printer was $435. These prices correspond to between a thirty-six and eighty-five percent reduction.

Disk Drives

Since 1970 when the average cost per byte of large capacity disk storage was about $0.0046, there has been a remarkable drop to $0.000037 per byte. This represents a

reduction of over ninety-nine percent. For the small capacity hard disks the price per byte is slightly higher, $0.000021 today and $0.0011 in 1980, but the reduction is still the same. Even for floppy disk storage the same reduction holds true. The price per byte dropped from $0.012 in 1980 to $0.00030 by 1986.

Memory

Internal memory prices, $6.64 per byte in 1970, have shown the most dramatic decrease. By 1976, this price averaged $0.108 and today averages $0.0035. This represents almost a one hundred percent decrease.

Graphics Terminals

A sixteen color, medium resolution graphics terminal with internal memory averaged $8,000 in 1983. Similar capabilities are available today for $450. If 64 or 4096 colors are desired the price is between $785 and $999. This is a ninety-four percent decrease.

Modems

The decrease in the price of a twelve hundred bit per second modem between 1970 and 1986 nears eighty-seven percent. The prices were $985 then and only $149 now.

Software Costs [2]

Although hardware prices have shown a steady, well-defined decrease over the past fifteen years, software prices have not followed the same trend. In fact, software prices have increased.

> Computer software has always been dismayingly expensive to write. In fact it is getting more and more expensive as computer usage grows and increasingly sophisticated software is needed. At the same time hardware costs are falling so that it is becoming feasible to use computers in many new ways; this makes the contrast between hardware and software costs more and more glaring [29, p. 1].

In 1973, U.S. companies spent over $10 billion on software. That figure accounts for more than one percent of the U.S. GNP [5, p. 48]. A report examining Air Force command and control requirements in the 1980's concluded "software costs are the tall pole in the tent -- the major source of difficult future problems and operational performance penalties." [5, p. 48]. For example, the Air Force expended $822 million for the World Wide Military Command and Control System by 1973. The software consumed over eighty-seven percent or $722 million of the total price [5, p. 48]. Also in 1972, NASA spent $200 million on software while only half that amount was spent on

---

[2] The figures quoted in the following sections were found in Datamation, Byte, and PC Magazine over the past fifteen years. Current figures were obtained by telephoning various vendors.

hardware. Software expenditures accounted for over sixty-nine percent of NASA's budget. In the decade of the 1960's, NASA spent over $1 billion on software for the manned space program [5, p. 48]. These figures are not confined to the DOD or NASA, but are also found in private industry. In 1972, IBM had spent over $200 million on the OS/360 software [5, p.48]. This trend of increasing prices has affected both systems and applications software.

Systems Software

In 1980, Honeywell Incorporated sold its FORTRAN compiler for $3,825, COBOL for $5,395, and a screen formatter for $1,540. Today these products can only be leased. Assuming a computer system life of five years, the price to use these products have increased to $12,780, $15,720, and $6,720 respectively. In 1980, the Burroughs Network Architecture software was priced between $12,750 and $21,000 depending on the hardware. Today the same software is available for $20,000 to $41,000. Software for the DEC PDP 11 and VAX environment has also seen the same increase. DEC's BASIC interpreter has increased by $2,250 in the past ten years. Additionally, the network software has risen from $1,500 in 1980 to over $3,500 today. DEC's DATATRIEVE, a query language, has increased over sixty-one percent since 1980 to a price of $7,260.

Microcomputer Software

The price increases have even affected software written for the microprocessor systems. A BASIC interpreter sold for $29 in 1976. By 1980 the price had doubled to $50 and by 1983 had increased to $279. Today the price is $350. A BASIC compiler has increased from $299 in 1983 to reach a price of $395 today. COBOL has increased twenty-five percent since 1983. Following this trend, a BASIC compiler has increased thirty-two percent in only three years. The BASIC interpreter has increased over 1100 percent since 1976.

Application software packages have not been immune to this trend either. In 1983, Supercalc was purchased for $189, Easywriter II $269, and Wordstar $279. Today, these packages are acquired for $395 (109 percent increase), $395 (47 percent increase), and $350 (25 percent increase) respectively. What could be a reason for this substantial increase in software prices? Examining the salaries paid in the data processing industry over the past fifteen years shows some interesting results.

Personnel Costs [3]

In 1970, the salary a manager of software development received averaged $23,000 per year and a manager of data communications $20,000 per year. By 1983, the communications manager could earn $50,000 per year and today a software manager receives between $41,000 and $66,600. Similarly, the salary for a systems analysis manager has risen from $25,000 in 1976 to between $29,000 and $56,700 in today's market. Managers of computer operations received $18,928 in 1976. Their salary today has increased to a minimum of $26,000 for a small company and to $67,000 for a large installation. Systems analysts currently receive between $29,000 and $42,500 which represents an increase from $9,776 in 1976. The position of programmer/analyst has increased from a range of $16,300 to $21,900 in 1976 to demand between $21,900 and $43,200 in today's market.

A systems software programmer with experience on IBM equipment was paid between $24,000 and $34,000 in 1980. Three years later this salary had increased to $40,000. Computer programmers received their share of the salary increases also. Entry level programmers averaged $22,980 in 1983 and with two to five years experience the salary

---

[3] The figures quoted on personnel salaries were found in Datamation, Byte, and PC Magazine over the past fifteen years.

reached $28,381. Today, these salaries range between $20,500 to $30,800 for the entry level position and $25,400 to $36,700 with experience.

Even the salaries paid to computer operators have increased since 1976. In that year, operators drew between $5,512 and $6,396. Ten years later their salary averaged between $15,600 and $22,200.

In conclusion, the greatest average yearly increase according to the data is the position of systems analyst. Their salary increased between 197 and 335 percent in ten years. Management positions rank second with increases between 37 and 254 percent. Programmers reached a maximum increase of 34 percent during the last three years. Operators showed the smallest, but still a substantial gain of 30 to 144 percent over the last ten years.

Analysis of Costs During the Last Fifteen Years

As shown in the previous sections, prices in the data processing industry have not remained steady over the past fifteen years. There has been a decrease of between thirty and ninety-nine percent across the entire spectrum of computer hardware products from memory to large mainframes. Unfortunately, the opposite trend has affected both the price paid for software and salaries. With salaries increasing between 30 and 335 percent during the

same period, the price paid for software has increased between 25 and 1107 percent. Since software development is a labor intensive job, the increasing salaries paid to software developers directly affects the price of the software.

CHAPTER III

REVIEWING MANAGEMENT AND ORGANIZATIONAL STRATEGIES

Every successful project needs adequate planning, effective controls, solid management techniques, and an appropriate organizational structure or it may be doomed to failure from the beginning. With this in mind, the requirements within these areas are covered in detail.

Planning

Objectives and Reasons

What is planning? "Planning means laying out what you (the manager) want to happen" [22, p. 101]. The plan is the means of determining the necessary resources and then allocating these resources. In addition, Bentley states that the plan must specify how these resources--people, money, hardware, or facilities--are to be controlled [2, p. 44]. The plan should specify who is responsible for each phase as well as for the overall plan. Plans by their nature are subject to change and should be flexible enough to respond to any deviations that may occur. Furthermore, Bentley says the plan should fit in with the project

standards and reporting methods of the organization [2, p. 44].

Contents

Any good plan should be broken down into a number of phases. Of course, the number of phases depends on the complexity, size, and duration of the project. Plans are best visualized by graphic means. Within each phase, the manager lists the activities, in chronological order, associated with that phase. Every phase and activity should be described in increasing detail. When appropriate, the plan should clearly state any potential problems and assumptions [2, p. 46]. For example, "before phase II begins all hardware must be installed and functioning."

Milestones

A milestone is any point in a plan where significant and clearly measurable progress has taken place. A milestone can include the completion of the programming, the installation of the hardware, or the completion of the design reviews. Using the fact that analysis is fifty percent complete is a very poor milestone. There is no way to determine when fifty percent has been reached until the task is completed. Each manager, from top to bottom, has responsibility for a set of milestones. According to

Metzger, managers at the lowest level should space their milestones at least two weeks apart. On the other hand, a month should be the maximum time between milestones. This will allow effective, but not over-burdening control on the project. At the completion of each milestone, the plan is checked. The manager checks the resources for effective utilization. If necessary, the manager reallocates these resources to meet any future milestones. In some situations, the plan may require changes [22, pp. 22-25]. Additionally, Donaldson specifies that keeping interactions and dependencies between milestones to a minimum will reduce changes to the plan [12, p.16]. Therefore, if one milestone is not completed on time it may not impact other milestones and require a major change in the plan. A survey of major software development projects revealed that milestones are used, but not as widely as needed. Only seventy-three percent of the projects recognized the completion of system definition as a milestone. For subsystem integration the figure rose to seventy-four percent [21, p. 120]. Refer to Table I for additional information.

TABLE I

MILESTONES USED IN MAJOR DATA PROCESSING PROJECTS

Phase	Percent Recognizing Completion As A Milestone
System Definition. . . .	73%
Requirements Definition.	83%
System Design.	92%
Module Design.	79%
Coding	94%
Module Test.	83%
Subsystem Integration. .	74%
System Integration . . .	85%
System Test.	94%
Operation.	66%

SOURCE: [21, p. 120]

Limitations

Most plans by their very nature are subject to change. In this respect, the manager should not overdo the paperwork generated during the planning phase. The plan should be complete before beginning a project; however, this does not mean the plan should contain detailed descriptions for the latter phases. This idea will be discussed in greater detail later in this chapter. Although network techniques like CPM and PERT will not be discussed, a plan should not be limited to only one of these techniques [2, p. 44].

Long-Range Plans

General Policy

General policy plans direct the future of the organization. They point the company in the right direction to meet their goals and aspirations in the coming years. Unfortunately, the Diebold Research Program in 1979 reported that eighty-seven percent of U.S. corporations lacked an overall information systems policy. Most of the companies surveyed grossed over $90 million in sales that year [17, p. 13]. There are several areas a data processing organization should include in its long-range organizational plans. First, new functional areas in which the company is interested should be closely investigated and planned accordingly. Next, the plan addresses major areas of hardware acquisition, facilities and staff expansion, and contingency planning. Areas like automation of the office or manufacturing require a solid commitment and long-range planning to insure a smooth and successful implementation. Fried suggests that the organization form a steering committee to assure involvement of all parties. This committee approves acquisition of all data processing equipment, short-range plans, and new projects to be undertaken. Additionally, the committee determines the

level of expenditures needed to meet these long-range plans [17, p.63]. Fried also says that a committee establishing the future plans of an organization must have consistency in its membership. No alternates should be allowed to replace permanent members as this may upset the group's consistency. The agenda should be set beforehand with advance copies of the agenda and any other presentations going to all members [17, p. 63]. Once the long-range plans are determined, the committee's job is to guarantee that all projects and plans help the company reach its future goals. The committee maintains and updates the goals as necessary.

Project Plans

Every project must have a long-range plan. This plan must enhance the long-range plans of the company and help the company reach a future goal. Fried states that long-range plans should be specific to eighteen months in the future. If the project is long enough, general plans are needed for up to three years [17, p. 237]. Long-range project planning incorporates both budgeting and resource allocation [2, p. 46].

Resource allocation includes everyone and everything needed to complete a milestone. A manager needs to know the usage and availability of each resource. This includes the daily/weekly availability of resources. For personnel especially, the manager must know how long he may keep the person [2, p. 47].

Short-Range Plans

Short-range plans are a detailed expansion of the long-range plan. Short-range plans project resources up to four weeks in the future. According to Donaldson, each job should cover two weeks with quick weekly reviews to keep the project on track [12, p. 20]. Furthermore, each job must have a well-defined start and end date. The person responsible for completion of a job must know this information. Consequently, the responsibilities of every person are also known [12, pp. 25-6]. The manager insures and verifies that these responsibilities are known, understood, and accepted. In general, plans should be flexible enough to fit into the organization. Plans should not be hidden, but available to everyone at all levels. This will let everyone know what lies ahead and gives them a better picture of the direction a project or organization is heading.

Controlling and Reporting

"Controlling means making sure it (the plan) does happen; and good communication (reporting) is a necessary tool for getting anything done" [22, p. 101]. The objectives of controlling include insuring quality and customer satisfaction along with keeping the project on schedule and within budget. Bentley specifies that the activities included in controlling are collection of project data and dissemination of this data to the proper levels [2, p. 50]. However, the most critical controlling function is solving problems. The manager must decide how to best handle each problem. Can the problem be corrected using available resources and without changing the plan or should higher levels of management be notified of the situation? "The process of controlling a software engineering project may well be the most talked about and least understood of all the project managers' functions" [4, p.56]. Lehman determined in a recent survey of software projects that seventeen percent had no control mechanisms implemented [21, pp. 123-24]. To aid a manager in keeping control of a project, Donaldson recommends several things that can be done. First, when issuing work to individuals, the supervisor must clearly explain and write down what is expected and when it must be completed. Second, the supervisor must check on the status of each

active job weekly. Then a problem can go unaddressed for at most one week. Last, after meetings, the administrative staff distributes notes to all attendees for clarification [12, p. 29].

Status reporting at all levels is a means of determining adherence to the project plan. Reports should include only measurable, predefined goals such as the milestones from the project plan. Top-level management does not need to know Module X has completed testing; however, the number of modules successfully tested would be something top-level management would be interested in knowing. Consequently, each report is tailored to the milestones of each particular level. If problems occur, the manager prioritizes these problems and lists possible solutions [22, p. 105]. Meetings are an excellent mechanism for dissemination of information and discussing solutions to problems; however, Brooks advocates that each meeting must have a stated objective and agenda to insure an effective use of time and energy [6, pp. 51-52]. To assist a project manager in gathering the data and distributing the reports a small administrative staff may be used. Metzger advocates using this staff to acquire computer time, perform keypunching, schedule training,

maintain documents, and manage contract changes [22, p. 85]. Even with a good plan, appropriate control, and effective reporting procedures, an organizational structure suited to the project must be employed.

Organizational Structures

The organizational structure is the means of implementing the plan and controlling the project. The organization must effectively perform the key areas of decision making and communications [2, p. 7].

Objectives and Reasons

Bentley suggests that there exist many reasons for having a formal organizational structure. First, and probably most important, lines of communication must be established. Internal communications incorporates the allocation of tasks and responsibilities to individuals. Along this vein, managers acquire the right person for each position. Each position has a description stating the responsibilities and skills required to fill the position.

Furthermore, Bentley advises addressing technical and administrative functions before determining an organizational structure and subsequently staffing the organization. The manager fairly allocates each function

to the technical and administrative areas with the interfaces and responsibilities between the two clearly defined. The more well-defined the organizational structure and the clearer the responsibilities, the less likely problems of authority will occur later in the project [2, pp.5-7].

According to Bentley, another objective of a formal structure is the definition of the overall management control of a project. Everyone is aware of their responsibilities; therefore, task delegation at all levels is possible.

External communications establishes the role of the user[4] or any other external entity. For example, this may include private consultants. As with the internal structure, the responsibilities of the user must be established. How the user will interface with the organization and when the user will be needed must be agreed upon [2, pp. 5-7]. Accordingly, where the user will be physically located must be decided. That is, will the users remain with their organization or move. Additionally, it should be clearly stated who has authority over the user. The data processing manager should have

---

[4] The user is the person or organization for whom software was developed. The user is any entity utilizing a data processing system. Users usually have knowledge of the functions their organization performs, but have little knowledge of data processing systems.

veto power over the users assigned to the project. The users may not fit into the data processing organization personality-wise or may be nonproductive personnel the organization is glad to let your organization have. These users should not be assigned to critical areas, but placed in analysis or testing [22, p. 137].

The final objective the organization provides is the necessary training to insure the continuing competence of all personnel. The training provided and the tasks assigned should help individuals move along the career ladder of the organization [2, pp. 5-7].

Limitations

The structure cannot solve all the problems of management. First, the organization of each project must be compatible with the overall structure of the company. Second, the project structure must be in harmony with project standards. Last, the structure must be tailored to the company and project [2, p. 7].

Roles

Most data processing organizations have a project manager, a user/analyst, and a committee called the users' group. The user/analyst provides assistance to the data processing staff when analyzing the requirements of the user's organization. Fried advocates training the user in

the data processing organization or placing an analyst from the data processing staff with the users for a period of eighteen months [17, p.37]. These transfers will aid in the design of new data processing systems.

The users' group, of which the project manager and the user/analyst are members, provides a sounding board for the analysis of the data processing requirements. The group monitors the project cost, progress, and resolves intergroup conflicts. Through the software project manager, usually the chairman, the group assures that adequate resources are available to meet project requirements [17, p. 37].

The software project manager is the most important position in any project. Not only must he plan, coordinate, and control all phases of the project, he also interfaces with the users. Evans states that responsibility for configuration management rests with the project manager; therefore, this individual approves all projects or changes to existing projects [14, p. 20]. Lehman found in a typical organization that the project manager is appointed by upper management and has ten years of data processing experience. Additionally, the manager has worked in the functional area for seven years. Furthermore, ninety-four percent of the managers had programming experience, but only fifteen percent had

experience as a lead or chief programmer [21, p. i21]. These three positions are usually found in each organization; however, the roles of other individuals and the organizational structures vary greatly.

Types of Structures

In the following sections, several organizational structures along with their advantages and disadvantages are discussed.

Job Shop

In the job shop approach, one manager handles all aspects of the entire system. This person has responsibility for analysis, design, coding, integration, testing, and finally implementation. Not only is this true of the manager but also applies to the programmers. The programmers design, code, integrate, and test all routines for which they are responsible [22, p. 71].

Metzger states that there are several advantages to this approach. First, everyone is involved in each phase. If people leave the project before it is completed, this may help continuity. Second, the control and responsibility for the system rests in the hands of a single person.

On the other hand, Metzger found several disadvantages to this approach. The most damaging is that the manager must be very competent both in technical and managerial skills. Without a well-qualified manager, this approach will fail. Next, since everyone is involved in every step and responsible for design through implementation, each individual's strengths are not fully exploited. Consequently, individual weaknesses become evident as the project moves through the different phases [22, p. 71].

Multiagency Group

Martin Marietta while developing software for the Viking Project, the satellite landing on Mars by NASA, used a multiagency approach. The group consisted mainly of supervisory and managerial personnel. Final authority and responsibility for the project rested with this group.

To aid the group in technical matters Martin Marietta established a software subgroup. The subgroup advised the multiagency group and recommended solutions in a variety of technical matters. The subgroup consisted of four design engineers, but they were not responsible for any software system. The subgroup coordinated the development of each system. Consequently, the subgroup held responsibility for integrating the various flight control, telemetry, science analysis, and tracking data systems [19, pp.193-196].

. Glass advocates using this approach when developing large, technically oriented software systems. The major advantage is the software subgroup. Having a separate group coordinate and integrate the systems helps produce a better quality product.

Conversely, this approach is not without its problems. The multiagency group did not follow the recommendations of the software subgroup on technical matters. Consequently, the multiagency group rushed the software development and changed the plan to integrate the software after delivery [19, pp. 193-96]. This proved to be a costly mistake. Lastly, no single person from the multiagency group had absolute authority or responsibility for the project.

In conclusion, this approach appears feasible if each group performs only its defined functions.

Separate Maintenance

Fried says that the establishment of a separate organization to perform software maintenance has few advantages. The only noteworthy advantage is that new software development personnel and plans are not burdened performing maintenance [17, p. 34].

Unfortunately, Fried finds that the advantages are outweighed by the disadvantages. First, maintenance personnel have less visibility and feel less challenged. Second, lower prestige and less opportunity for advancement also plague maintenance personnel. As a result, there is a higher turnover rate in personnel [17, p. 34].

Automated Data System (ADS) Manager [5]

Several organizations within the Air Force use the ADS manager approach. The ADS manager holds responsibility for an Automated Data Processing System (ADPS). An ADPS consists of all the resources necessary to perform a specific function. The ADPS will be discussed in greater detail in Chapter IV. Figure 2 shows the structure of this organization. The ADS manager is responsible for all aspects, hardware and software, of an entire ADPS.

One advantage of this approach is that software is considered as important, if not more important, than hardware. Each ADPS has a single point of contact, namely the ADS manager. In addition, quality control and testing, very important phases in software development, are performed by an independent organization. Lastly, the same organization that designed and coded the software performs the maintenance.

---

[5] This information about the ADS manager strategy was obtained while working as an analyst in the Air Force.

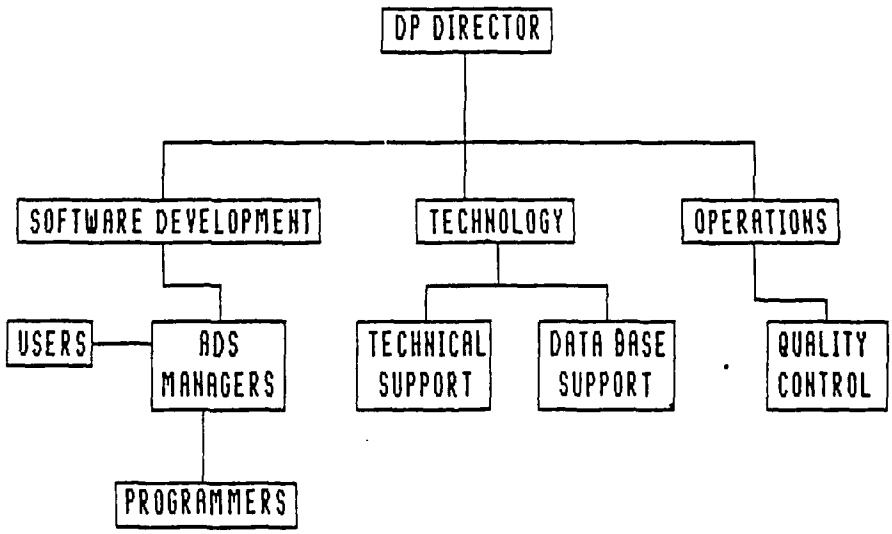

Figure 2. ADS Organizational Structure

Unfortunately, this approach does have several disadvantages. First, the ADS manager has too much responsibility. These responsibilities include interfacing with the users and other divisions within the organization. Additionally, the manager has to perform analysis and design while managing projects through coding, testing, and implementation. The systems software division designs the data base. Since this is an applications software function, it should be performed by the applications software division. This problem emphasizes the major disadvantage of this structure. The ADS manager has the responsibility but not the authority to assign tasks to personnel in other divisions. The ADS managers spend most of their time writing letters assigning tasks to other divisions or determining the status of tasks in the other divisions. Finally, quality control did not become actively involved until the latter phases of each project.

Functional Responsibility [6]

In this type of organization, each programmer/analyst assumes responsibility for an entire functional area. This area may be accounting, payroll, or inventory, for example. Refer to Figure 3 for an example of this

---

[6] The information presented for the functional responsibility approach is from personal experience while working as a programmer/analyst for a private company.

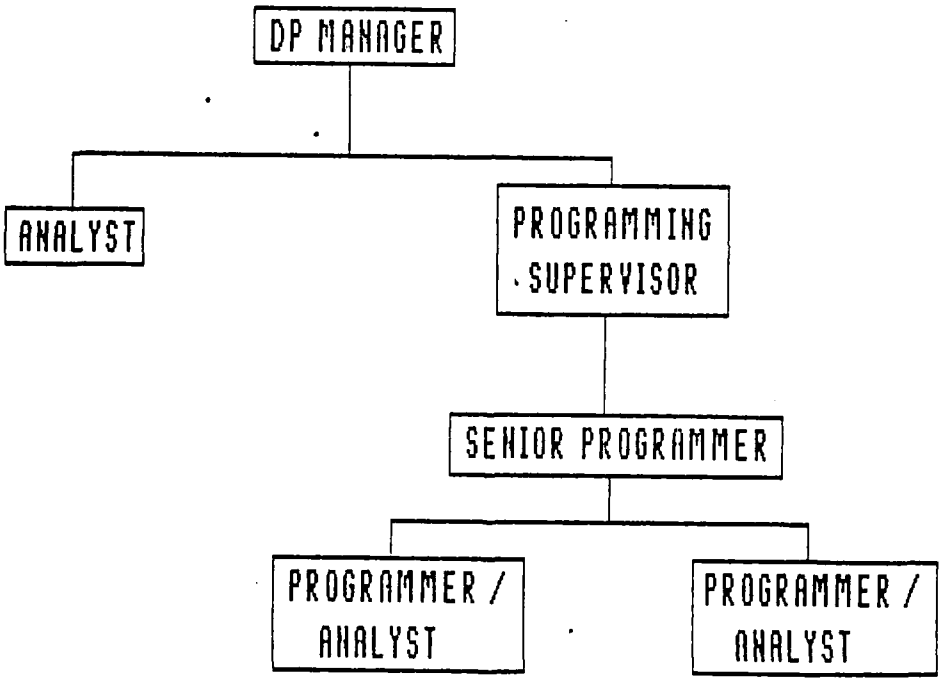

Figure 3. Functional Responsibility

organization. The programmer/analyst becomes the point of contact for each system. The programmer/analyst interfaces directly with the users for routine problems and assistance. To provide control over the functional areas a senior programmer assigns the tasks to the programmer/analyst. For most routine tasks the user will work with the senior programmer to establish deadlines and priorities. A programming supervisor and a systems analyst provide expertise and assistance to the programmer/analyst when required.

This strategy works well with a staff of ten or less and fewer than six functional areas. Exceeding these limits will cause the lines of communication to become unmanageable. Each programmer/analyst becomes very familiar with his particular system. The programmer/analyst also can slowly learn the techniques of design and analysis by watching the system analyst and training on small projects. In addition, a single point of contact is available to the users. Furthermore, the programmer does not have to go through channels to talk to the user. Finally, there is a clear delegation of responsibility. Consequently, excellent opportunities for training and career advancement exist within this structure.

Unfortunately, the programmer/analyst may not see the big picture of the company. Even in a small organization, the programmer/analyst may never become actively involved in the other functions of the organization. Since the organization is small, quality control, documentation, and configuration management are not addressed. These problems can be overcome by hiring several additional people. On the other hand, if the current staff will not be burdened with these additional duties, no change in the organizational structure is needed.

Centralized Design [7]

If the same function is performed at several locations, the software can be developed at a central design site. Consequently, the hardware at each site must be compatible. Refer to Figure 4 for this organizational structure. This structure will build upon the organization in Figure 2. Although the title is the same, the responsibility and authority of the ADS manager have been drastically altered. The ADS manager still interfaces with the users; however, the manager is now chairman of the users' committee. As chairman, the manager, along with the users' committee, are responsible for long and short-range

[7] The information presented for the centralized approach is a combination of personal experience and the opinions of other people about their organizations.

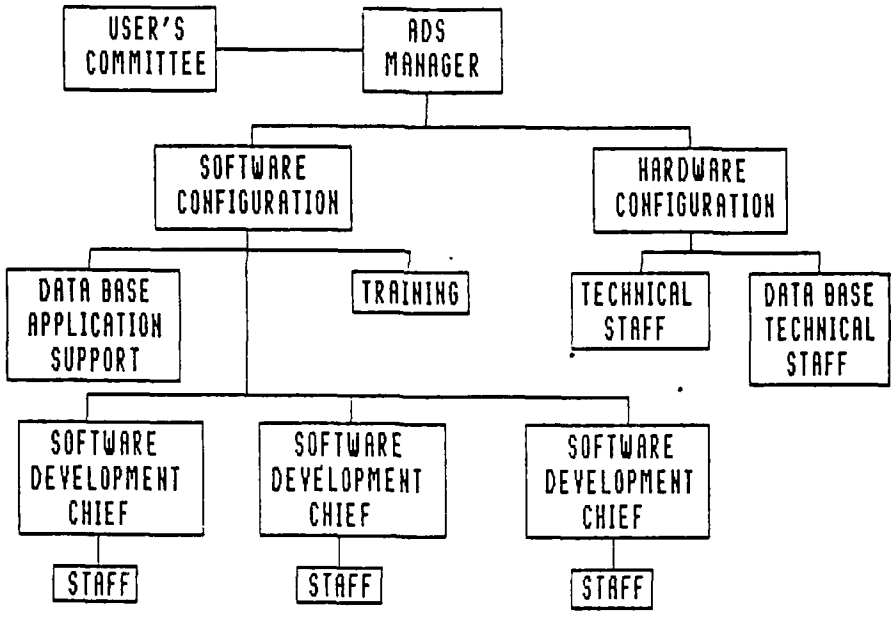

Figure 4. Centralized Software Design and Hardware Configuration

plans. Along this vein, the manager approves all requests and prioritizes these requests to meet the plans. In the capacity of chairman, the manager calls periodic meetings of the users' committee to review, update, and approve changes to the plans. Additionally, the committee approves or disapproves all new projects. The approved projects are then incorporated into the plans. Fried advocates that the manager maintains the current status of all resources and reviews the plans to determine the impact upon these resources. When more resources are needed to accomplish the plans, the manager advises top management of the alternatives and makes a recommendation [17, p. 50]. In the managerial area, the manager establishes and enforces standards, reviews the status of all projects, and insures the effective use of all resources [17, pp. 49-51]. The manager coordinates the configuration between hardware and software to insure compatibility. The manager has complete and absolute authority over the ADPS. This includes assigning tasks to the users' committee and both configuration managers. Although not directly involved in configuration management or software development, the ADS manager should have some experience in these areas. Besides having a technical background, the manager should be an effective communicator because of his role as chairman of the users' committee.

In addition to the responsibilities already stated for the users' committee, they may aid the ADS manager or even the software development chiefs in analysis and design. The users' committee becomes extremely important when centrally designed systems are acquired [17, p. 59]. The ADS manager may receive new tasks or changes to the current configuration from the users' committee or from the site managers. In any event, the users' committee must approve all changes. The users' committee need not be a standing committee, but may be called together when needed.

Quality control and testing remain in a division independent from the ADS structure. On the other hand, the users' committee aids in the analysis, development, and implementation of the software test plan.

The software configuration manager is responsible for all current and proposed software systems. In the area of coordination and long-range plans, this manager meets periodically with the hardware configuration manager to determine the impact of the long-range plans on their current resources. With the aid of quality control, the manager oversees the development and implementation of systems test and acceptance plans. The software configuration manager performs system analysis and design with the aid of the software development chiefs. Once the system design is completed, the manager assigns the

development of the subsystems design to the chiefs. The manager then approves the detailed subsystem designs of the chiefs. In the next step, the software configuration manager and the chiefs establish the milestones for the development of the software. Finally, quality control, the software configuration manager, and the appropriate chief integrate the software subsystem.

The hardware configuration manager develops plans for the implementation of the hardware at all the sites. In this area, the manager interfaces with computer operations in regards to facilities and actual installation. The hardware configuration manager reviews and recommends approval or disapproval of all hardware modifications at the sites.

The technical staff directly supports the sites and the software development chiefs. Furthermore, the technical staff keeps the hardware configuration manager informed about new technologies and how, if possible, to effectively utilize these new products. The number and variety of systems software products is overwhelming. For any given organization to keep abreast of the available products, the organization should assign several specialists the responsibility of keeping track of these new products. These products should be reviewed and a list updated yearly. The technical staff tests and installs new

systems software and utility programs. Additionally, any new hardware is tested at the design site before installation at the users's site.

The data base technical support staff insures that the data base management system is operating correctly and efficiently. In this regard, the data base technical support staff maintains the data base as well as keeps an audit trail of all data base activity.

The software development chiefs assume responsibility for assigning responsibilities to their personnel. These responsibilities may include designing, coding, testing, or integrating the modules within a subsystem. The chiefs oversee all phases in the development of their subsystem including reviews and walk-throughs. Bentley uses the software development chiefs as the key individuals for the enforcement of standards. The software development chiefs establish deadlines, lines of communication, and responsibilities for the individuals within the subsystem. The chiefs establish the structure of the team, the skills required, and the training necessary to complete the subsystem [2, p. 12]. To aid the chiefs in the development of software a user/analyst, supervised by the chief, should be assigned to each subsystem. The user/analyst interprets and elaborates the requirements of that subsystem [2, p. 12]. Consequently, the user/analyst must

have an excellent knowledge of that subsystem and must be located within the software organization. The user/analyst has no authority to accept software or change any requirements.

The data base application support staff provides programming support and analysis on data base matters to the chiefs. The data base applications support staff is responsible for developing the data base schemas[8] and establishing data base standards.

Probably the most overlooked area of software development is user training. A separate division is responsible for developing and implementing the training. This division develops the plans and schedules along with the users' manual. Systems and hardware familiarization, operator training, and user training must all be addressed. The user and familiarization training may be performed by the training staff at the users' sites; however, operator training might be performed at the design site prior to arrival at the user sites.

In addition, the ADS manager has authority over the sites where the software and hardware are installed. The site managers have control over their installation, but are responsible to and are supervised by the ADS manager.

---

[8] The data base schema is the logical organization of the data base and the methods needed to access this data.

Refer to Figure 5 for the structure of the users' site. The site managers are also chairmen of their users' committee and make recommendations to the ADS manager. Additionally, the site manager researches, documents, and recommends solutions to the ADS manager when problems occur. Since each site may have a slightly different hardware configuration, the site manager must keep the requirements of his site current.

A single division performs the job of hardware and software configuration management. This division implements all software releases and recommends hardware updates to the site manager as the software changes. Lastly, this division acquires and distributes documentation.

The advantages of a centralized approach to software development are many. First, this approach requires fewer management and data processing personnel. For example, if five sites developed their software independently from each other, each site would have to duplicate the personnel required for one site. Additionally, with a centralized approach a common data base and data dictionary[9] could be developed. Next, standards are easier to establish and enforce. Resources are more efficiently utilized because

---

[9] The data dictionary is a description of the data names in a system, along with their length, where used in each module, and how used.

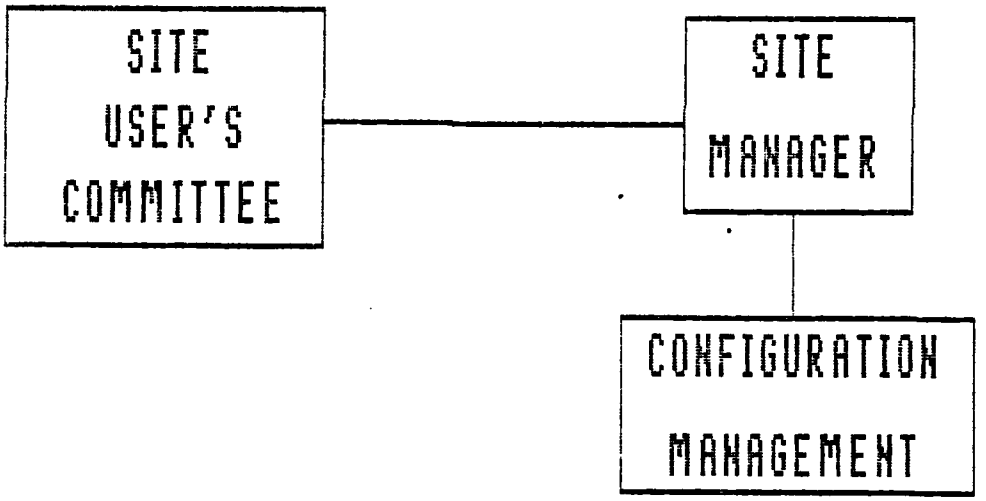

Figure 5. User Site With Centralized Design and Hardware Configuration

redundant software development is eliminated. Staffing the project with skilled people and retaining these people becomes easier. Fried states that one reason for this effect is the ability of personnel to gain valuable experience while working on projects of this size and magnitude [17, p. 40]. Additionally, since compatible hardware is acquired, reduced costs may occur by purchasing the equipment in larger quantities. Management is centralized with a single manager responsible for both software and hardware. The ADS manager has the responsibility along with the authority over all aspects of the system life cycle. Furthermore, each level of management has clearly defined roles and a good delegation of responsibility exists between these levels. Since each user site can maintain a different hardware configuration, the hardware is tailored to the software. This is the correct way to develop software. Because the users' committee at the design site constantly updates the requirements of the system, only a short lead-time is necessary when the system requires replacing.

Similarly, disadvantages also exist for this strategy. First, and most important, the requirements of all sites must be addressed and satisfied. The product must be thoroughly tested and highly reliable if it is to be used at all the sites. The ADS manager becomes the

most important individual in this organization and must know and perform his duties almost flawlessly. When errors occur or modifications are necessary to the software, response time may be slower compared to other approaches. Also, additional traveling expenses for the users' committee and the training staff must be included in the overall cost of the system.

Decentralized Design [10]

Decentralized design with centralized configuration management is appropriate for developing software for several sites having different requirements. Under this strategy, similar requirements are grouped into functional areas for centralized control. Configuration management remains centralized to insure compatibility between all the sites, enforce standards, and assure no duplication of effort occurs at the sites. Because of the centralized management and enforcement of standards, software developed at any site can be easily transported to any other site. Again, this strategy uses the ADS manager as the central point of authority. The manager's responsibilities are a little different than those of centrally designed software. Figure 6 depicts this organizational structure.

---

[10] The information presented for the decentralized design approach is a combination of personal experience and the opinions of other people about their organizations.

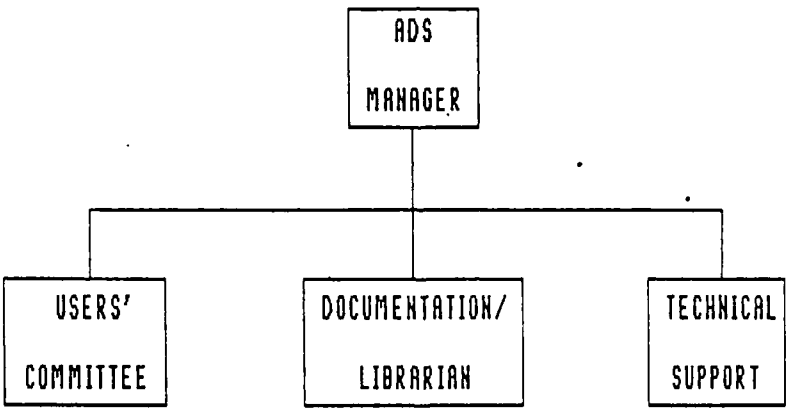

Figure 6. Decentralized Software Design With Centralized Configuration Management

The ADS manager remains the central authority for the entire ADPS; thus, the manager approves all projects. Additionally, the ADS manager along with the users' committee perform the same functions as with centrally designed software. On the other hand, the ADS manager delegates more responsibility to the site managers where the software is actually developed. The ADS manager will not become involved in analysis, design, or configuration management. The manager through periodic meetings with the site managers will consolidate similar requirements from different sites into a single package. This package will then be assigned to a single site for design. The other sites involved will aid in the design; however, their major responsibility will be quality control and testing. This approach should insure a better quality product.

Also in this strategy, the users' committee will develop and update the long-range plans for the entire ADPS. Likewise, the committee consolidates the requirements of each site. In the same respect as the manager, they aid in approving all projects, but do not become involved in the design.

To assist the ADS manager and users' committee, a technical staff provides assistance and recommendations when needed. The technical staff tests any new systems software before installation at the sites.

Since each site has different requirements and therefore different software, a documentation/librarian is needed. The librarian keeps copies of the long-range plans of the ADPS along with the requirements of each site. The librarian also maintains all documentation concerning standards and procedures. Together with the aid of the configuration manager of each site, a description of each system and subsystem is retained. The users' committee utilizes these descriptions to determine if a new requirement can be performed using existing software.

The organizational structure used at the design site of centrally designed systems will be incorporated at the development sites of decentrally designed systems. All responsibilities will essentially remain the same. Refer to Figure 4 to review this structure. The ADS manager in Figure 4 now becomes the site manager. The site manager still retains complete authority over the development process, but must receive approval from the ADS manager on all projects. Another change is in the function of the training staff. The training staff still provides operator and user training, but only for its site. The hardware and software configuration managers have the additional responsibility of updating the documents maintained by the documentation/librarian at the central site.

The advantages of this strategy are numerous. First, since the sites develop the software, the development staff is more attuned to the needs of the users. Consequently, the software is optimized to the needs of the particular user. Furthermore, the organization can respond quicker to changes in requirements and priorities. Fried finds using this approach that the users have tighter control over the software and a more personal involvement in its development [17, pp. 45-6]. As with the centrally designed software, quality control is separate. Additionally, all the organizational and managerial advantages inherent in the centrally designed strategy are also in this decentralized strategy. Similarly, the hardware still remains tailored to the software requirements.

On the contrary, documentation and standards are harder to enforce between multiple locations. Also, since the software is developed at each site, personnel costs will increase. According to Fried, this in turn will increase the software costs [17, p. 55]. Besides these problems, software compatibility between sites may be harder to maintain. Finally, since each site performs a slightly different function, determining the sites that belong under an ADPS may be difficult. Placing a system into the wrong ADPS can increase its software costs since the hardware may not be optimally suited for that function.

Chief Programmer

The chief programmer approach in developing new software has been effectively used by Systems Development Corporation (SDC) and Computer Sciences Corporation (CSC). The main feature of this approach is an experienced programmer, the chief programmer. A small team, ten or fewer, of programming assistants supports the chief programmer. SDC claims that this strategy increases productivity by a factor of ten over line organization [19, p. 197]. A medium sized group of analysts and user representatives performs the analysis. Refer to Figure 7 to review this strategy. According to Baker, when the design is initiated, the chief programmers are integrated into the group; when the design nears completion, the backup programmers are brought into the group. At the same time, the specializations and skills needed for the programming teams are determined and training is initiated. When the design is approved, the chief programmer and backup will be very familiar with the requirements and the programming teams will have just completed the necessary training. To supplement the staff an analyst from the design group may assist the chief programmer [1, p. 60].

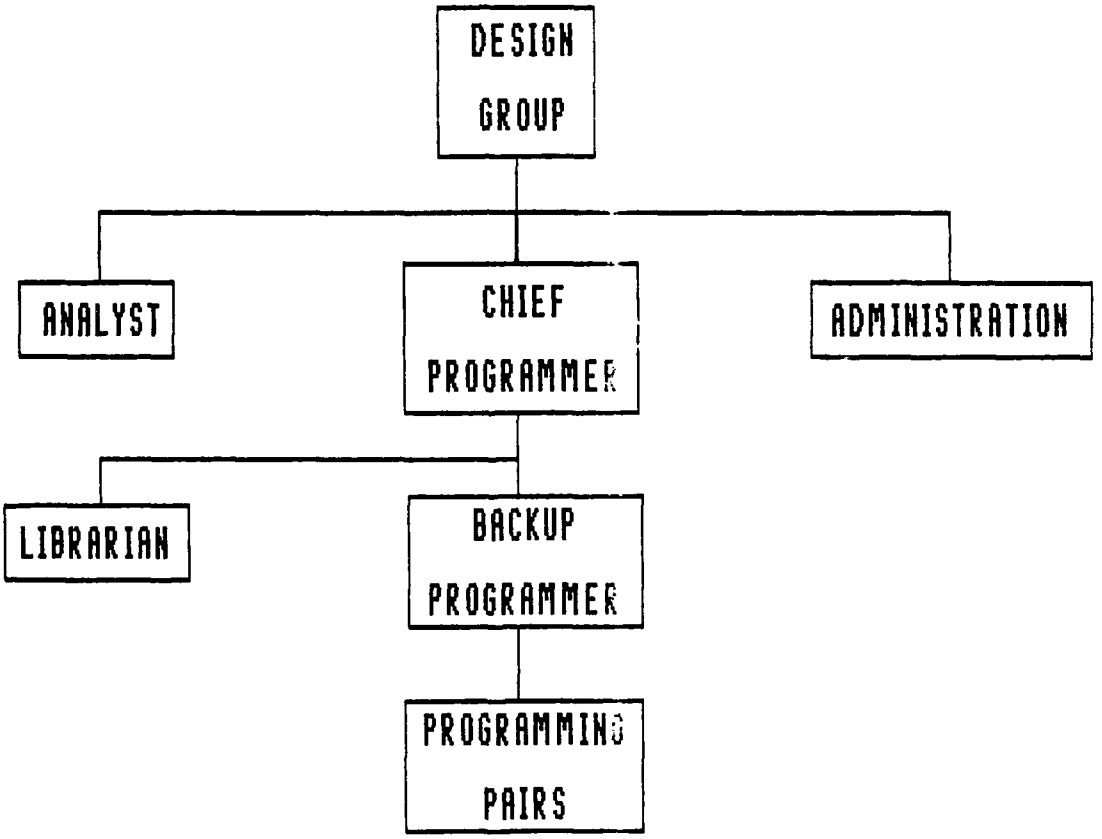

Figure 7. Chief Programmer Organization

Glass states that the chief programmer handles responsibility for the overall detailed design of the subsystem and codes the main routines. Although freed from administrative details by a support staff he chief programmer reviews and critiques the work of the programming teams [19, p. 198]. The chief programmer has a technical background and all programmers report to him [1, p. 59].

Baker utilizes the backup programmer as a sounding board for the work of the chief programmer. The backup performs research for the chief and tests the routines written by the chief. Likewise, the chief tests the programs written by the backup [1, p.59].

The librarian keeps the status of all tasks. Similarly, the librarian insures that all documentation is completed according to standards [19, p. 198].

There should be about four or six programmers [1, p. 58], working in pairs to critique each others' work [19, p. 198].

In large development projects, SDC recommends using a chief architect to oversee several teams. Following the same idea, SDC also recommends not overburdening the chief programmer with administrative details and paperwork [19, p. 203]. CSC solves this problem by using a department manager to perform these functions. This manager should

oversee two to three chief programmers [19, p. 204]. Martin Marietta refined the concept of the programming team. The two members were a programmer and an engineer. The engineer developed the requirements and tested the software. Concurrently, the programmer designed and developed the software to meet these requirements [19, p. 205]. The engineer split his effort equally between developing requirements and testing the software. On the other hand, the programmer expended two-thirds of his effort to design and code the software. The remaining effort was put into testing and maintenance [19, p. 207].

The major advantage of this approach is that two people are very familiar with each subsystem and each module. Furthermore, since the programmers are trained and experienced in the skills required to complete each module, better quality code is developed in a shorter period of time [1, p. 58].

The major disadvantage is finding people skilled in certain areas of specialization. Subsequently, when the project is completed, these people may no longer be needed. Lastly, each task requires two individuals.

Consultants

A consultant is anyone or any organization used on a project and having no vested interest in the approval or completion of that project. In other words, consultants come from an independent and autonomous organization. There are many instances when a consultant may prove invaluable to an organization. Fried states that a consultant may have skills hard to find in the open market. Additionally, a requirement or position may not justify hiring a full-time specialist. The consultant may bring a new and objective viewpoint to a project. Similarly, this independent viewpoint may carry more impact with top management [17, p. 285]. Consequently, consultants may free staff from participation in new projects so they can perform their daily and necessary duties [18, p. 130].

When an organization elects to hire a consultant, several steps should be followed in selecting an appropriate consultant. First and foremost, Fried suggests that the organization clearly defines the area in which a consultant is needed. Next, the organization determines the skills required to complete the task. Lastly, they

specify what type of end product the consultant will produce and how long the consultant has to produce this product. When using a consultant for more than a couple of days, methods and frequency of reporting are also desirable [17, pp. 289-90].

Once these questions have been answered, Fuchs advocates contacting a reputable consultant association [18, p. 130]. There are four major associations and each has a referral service [20, p. 130]. After obtaining the names of the appropriate companies, the organization contacts these companies to get the names of individuals who satisfy its requirements. These individuals are interviewed to determine their ability to help your organization [18, p. 130]. During each interview, Frankenhuis suggests obtaining a list of other clients each consultant has worked for in the past several years [16, pp. 136-7]. After completion of the interviews, the organization contacts these clients to determine the accuracy of its information.

After the consultant is selected, the organization should assign a liaison to the consultant. The liaison obtains information and answers any questions the consultant may have. Prior to the consultant's arrival, the liaison determines and prepares any information and

documents the consultant needs. A letter of authority from top management will inform the organization of the consultant's presence and responsibilities [17, p. 294].

The contract between an organization and a consultant must be a clear and concise agreement. Fried says that the contract must state how the consultant will interface with the organization and how control will be maintained. A start and end date should be specified along with a checklist of deliverables. Charges and terms of the agreement should be fixed-cost. Since consulting firms are always interested in obtaining skilled personnel, the contract should forbid the consulting firm from hiring employees of the client. Next, any information the consultant obtains while working for an organization must remain strictly confidential [17, p. 293]. Lastly, the client must insist on no substitutions of personnel.

When a consultant has finished the required task, the organization performs an evaluation. This evaluation compares the results from the consultant against the list of deliverables in the contract. If all the requirements were satisfied, this firm can remain eligible for bidding on future contracts; however, if an unsatisfactory job was performed, this firm must be removed from consideration on future contracts [18, p. 132].

In the beginning of the section, reasons to use consultants were presented. Unfortunately, dangers also exist. According to the author, the morale of the permanent staff may decline. The staff is not directly involved in the new project and feel their abilities are not fully utilized. Next, conflicts of interest have always been a problem when using outside agencies. According to Kennedy, the organization must strive to remain independent of relationships other than professional with consultants [20, p. 121]. Similarly, a consultant may lose objectivity as these relationships grow. Kennedy recommends that organizations not always use the same consulting firm [20, p. 117].

User Involvement

With the growth of the data processing industry over the past decade, the involvement of the user has become critical for successful completion of most projects. Boehm found that thirty percent of today's work force deal with computers on a daily basis. On the other hand, only fifteen percent understand how the computers work [5, p. 57]. Figure 8 shows the growth in the use of computers since the mid 1950's. First, users should be involved in all phases of the project except the actual coding and programming [14, p. 175]. Evans found that when the user

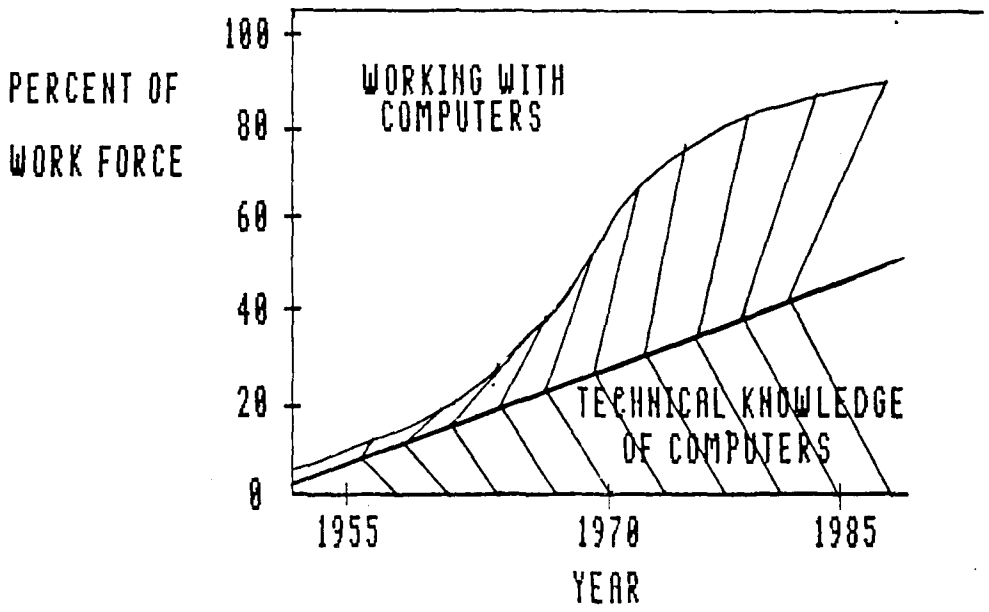

Figure 8. Growth in Computer Usage Since 1955 [7, p.57]

takes part in the development of a project, individual requirements are listed and clearly traceable. Additionally, these requirements serve as the basis for final acceptance testing [14, p. 38]. To eliminate problems that may occur in the latter phases of a project, the user should be deeply involved in the critical parts of the system. Furthermore, Donaldson advocates recording all agreements and requirements between the user and the data processing staff [12, p. 23]. Finally, acceptance of any system rests solely with the user. The project is not completed until the user accepts the system. Foss found that if a user has been involved in the development of a system and has confidence in the data processing organization, the user may agree to a longer development period. If the user is anxious to get the new system operational, Foss suggests that the data processing staff phases the delivery of the software and makes the users feel they are in control of the project and the design of the system. To achieve this the data processing staff should develop and present to the user several workable solutions. For each solution Foss says to list the good and bad points as well as the manpower, cost, and estimated completion date. Having this information in hand, let the users pick the design they feel the most comfortable with [15, p. 2].

Project Staffing

As with most projects, the staffing requirements of a software development project increase as each phase is completed. Since new personnel added to the project must be trained, the staff of the Massachusetts Institute of Technology recommends that these personnel increases should not exceed twenty percent each year [6, p. 46]. The staff at Bell Labs suggests personnel increases should not exceed thirty percent. Increases greater than this will inhibit both communications and the informal structure of the organization [6, p. 46].

In this chapter, several organizational structures along with their advantages and disadvantages have been presented. Using this information, selection of an organizational structure appropriate for Air Force data processing projects will now be addressed.

CHAPTER IV

SELECTING AND IMPLEMENTING AN ORGANIZATIONAL STRUCTURE

Before selecting an organization's management structure, the hiring policies and practices of the organization must be understood. For example, the Air Force cannot advertise for a communications specialist as can a private company. Furthermore, the size of most Air Force data processing projects is large with many interfaces to other systems. Keeping these facts in mind, the centralized and decentralized organizational structures discussed in Chapter III will meet the needs of the Air Force or even the DOD. The Air Force must implement one of these two structures at all data processing locations. These organizational structures are depicted in Chapter III as Figures 4, 5, and 6. Additionally, the planning, centralizing, and implementation phases of the Phase IV Project, the standard personal computer acquisition, the Electronic Warfare Integrated Reprogramming System (EWIR), and the wargaming systems are reviewed in this chapter for their adherence to these management approaches. The Phase

IV Project and the standardized personal computer acquisition followed the correct approaches; however, the Electronic Warfare Integrated Reprogramming System (EWIR) and the wargaming systems require a more standardized approach.

Reasons for Selection

Since all Air Force projects fit into one of these two organizational strategies, the Air Force does not need any other organizational structure for managing data processing systems. As stated in the previous chapter, the major advantage of these approaches rests with the centralization of configuration management. Using this approach, once an ADPS chooses between the centralized or decentralized strategy for its highest managerial level, the ADS manager can mix the two strategies in the lower levels. For instance, the top level may be centralized; however, if a user site (lower level) receives inputs from three separate locations, these three locations may be candidates for decentralized design. If these three locations each perform a slightly different function, the software at these three sites must be developed using a decentralized approach. The ADS manager at the top-level organization

maintains control over and responsibility for this project. After selection of the appropriate strategy, the Air Force must place all the current data processing systems under one of these strategies.

Using Current Policies and Procedures

The Air Force currently maintains a list of all functional data processing areas. This is the ADPS list. There are currently seventy-seven ADPS's. For example, ADPS B1 encompasses the

> USAF Academy Data Processing Support--This system supports the mission of the USAF Academy as related to the administration and management of student and faculty activities. This system also provides data processing support to student/faculty research projects [10].

Gunter Air Force Station in Alabama currently maintains this list, which forms the basis for standardization of Air Force data processing systems. Furthermore, several data processing systems have de facto standards that can be incorporated into their new systems. This is the case with the wargaming systems discussed later in this chapter.

Transition Plan

The following sections introduce a plan needed for the transition to this new approach. The plan will not incorporate a timetable because the number of systems and the amount of coordination necessary to implement and

complete this project is hard to estimate at this time. It may take fifteen years to place all Air Force data processing systems under this strategy.

Planning

The first phase of this project is planning. In this case, planning for the transition to the new approach needs addressing. First, an organization obtains authority to maintain and update the ADPS list. The Air Force should locate this organization at Scott Air Force Base because the headquarters for communications, of which data processing is a part, resides there. Final authority in assigning systems to the appropriate ADPS or, if the situation warrants, creating a new ADPS rests with this organization. At the same time, this organization prepares guidelines to justify and document the requirements of current or proposed systems. With these two steps completed, the organization reviews the current ADPS list and updates it when definitions and specifications need clarification. Additionally, the organization prepares more guidelines to help all Air Force organizations place their systems into an ADPS or petition for creation of a new ADPS. The Air Force must develop and enforce new regulations to insure proper utilization of the ADPS list

when developing new software. Consequently, all new requirements are checked against the ADPS list to determine if existing software can satisfy any of the requirements. Adhering to this procedure the Air Force can eliminate a large amount of duplicated effort. Finally, this organization must properly maintain and update this list. To aid in a yearly review of this list a point of contact at each site developing software may prove helpful. Currently, the ADPS list contains only the ADPS code and description. To be effectively used the Air Force must expand the list to include basic hardware configurations, places installed, points of contact, phone numbers, and descriptions of the software.

With these steps completed, the design sites can make inputs to the list. First, the Air Force sends the ADPS list along with the guidelines for placing systems into an ADPS to all commands, separate operating agencies, and private government contractors. These organizations will assign an individual to distribute the forms to all software development staffs within their organizations. This person maintains responsibility for returning the completed forms to the ADPS organization. Each development staff places all software systems into an ADPS, including descriptions of both the hardware and software. If a system does not fit into an ADPS, the descriptions are

still necessary and the ADPS organization places these systems into an ADPS.

Once all the forms have been received by the ADPS organization, the organization reviews them for accuracy, clarity, and completeness. If necessary, the ADPS organization will assign some systems to a different ADPS or create new ADPS's. After completion of this phase, the commands receive the updated list for distribution to the appropriate organizations. The Air Force can now use this list to standardize all current and future data processing systems.

Centralizing

With the completion of the ADPS list, the Air Force can initiate centralization of each ADPS. First, each ADPS must locate a central administrative site, containing the ADS manager, for the particular ADPS. The Air Force should strategically locate the administrative sites in regards to all locations utilizing each ADPS. Furthermore, these central sites require facilities for the necessary staff and hardware. Next, the Air Force selects the ADS manager. The requirements of the ADS manager include a background in data processing management and experience in the ADPS. The ADS manager then selects members for the users' committee from the sites utilizing each ADPS.

Having established this committee, the ADS manager assigns each member the task of determining the current and future functional requirements at each site. The individuals on the committee document and justify these requirements. The ADS manager and the users' committee will consolidate these separate requirements into a concise design specification. Consequently, the users' committee must not overlook any requirements. To help the members of the ADPS users' committee determine this information a local users' committee may be established.

Upon completion of the analysis of requirements, the ADS manager calls the users' committee back together. Over a period of many meetings they determine which requirements are truly justified. Accordingly, the committee consolidates the functional requirements into a single document. Reviewing this document, the committee determines the management strategy, centralized or decentralized, for the development of the software. Next, the users' committee determines if any current software system meets these requirements. If software does exist, the committee determines the hardware configuration best suited for the software. On the other hand, if appropriate software does not exist, the committee specifies the hardware configuration that fits the functional requirements and begins designing the new system. Having

now completed both the software and hardware requirements a request for information may be sent to vendors and contractors.

The Air Force has standardized the data processing systems utilized at most Air Force bases. This standardization was the Phase IV[11] Contract. The first function, the Standard Base Supply System provides an automated inventory control for supply accounts. Next, the Phase IV System consisted of the Customer Integrated Automated Purchasing System. This purchasing system automates the solicitation, acquisition, and delivery of base commodities. Additionally, Phase IV incorporates two personnel systems. One pertains to Air Force civilian employees while the other handles Air Force military personnel. Furthermore, the Phase IV System includes accounting and pay systems.

Before awarding the Phase IV Contract, these six systems operated on three separate hardware systems. These hardware systems consisted of the Univac 1050-II, Burroughs B3500/3700/4700, and Honeywell 725G. The contract required compatibility and expandability between all hardware and software. In addition, the contract specified each base

---

[11] The information presented on the Phase IV System is from U. S. Air Force Contract No. F19630-81-D-0002, Functional Performance Specifications--Part I; Automated Systems Program Office, Directorate of Contracting/PSK, Gunter Air Force Station, Alabama, 2 September 1982.

could configure the hardware to meet its requirements. Finally, the contract stated procedures and training must be standardized.

In reviewing the requirements of the Defense Mapping Agency, Air Force Reserve, Air National Guard, and the Air Force, the design staff proposed three basic hardware configurations. The X1 hardware system replaced the Univac 1050-II and performed the supply and purchasing functions. The X2 hardware system replaced the Burroughs B3500/3700/4700 and handled the two personnel systems, the accounting system, and the pay system. For locations performing all six functions, the design staff established a regional data center. This hardware configuration was called X3. For each hardware configuration, the contract specified the minimum equipment required to satisfy the workload. For example, the X1 system has five workload categories. For workload category 1, the contract required 200 kilobytes of memory, 65 million bytes of disk storage, three tape units, a line printer, card reader, and card punch. For workload category 5, the memory increased to 500 kilobytes and the disk storage increased to 145 million bytes. Each location specified its workload category for each system and from these requirements the contractor determined the hardware needed at each location.

The contract stated the basic requirements for each piece of peripheral equipment. These basic requirements pertained to memory, disk storage, card readers, card punches, line printers, tape units, desktop printers, keyboards, and visual display units. For example, a visual display unit must display a minimum of twenty-four lines of eighty characters per line. Also, the display area must be at least seventy square inches.

Another data processing requirement that the Air Force has standardized is the acquisition and use of personal computers. With the increased use and availability of personal computers, the Air Force saw the need for a standardized personal computer system. Prior to 1983 each command within the Air Force had a separate contract for acquisition and maintenance of personal computers[12]. This new contract specified a basic hardware configuration of one 8085 processor (8 bit), one 8088 processor (16 bit), four expansion slots, 128 kilobytes of memory, one twelve inch monitor, two serial ports, one parallel port, two floppy disk drives, and one keyboard. The software included CP/M-85 operating system, BASIC-80 interpreter and manual, diagnostic software and manual, and a technical

---

[12] The following information is found in Contract No. F19630-83-D-0005, Air Force Computer Acquisition Center (AFCAC) Project 229, AFCAC, Hanscom Air Force Base, Maryland 01731, 03 October 1983.

reference manual. In determining these hardware and software requirements, the contract required the use of currently available and tested products. With no requirement for the development of any new hardware or software, the Air Force substantially reduced the price of the contract. The Air Force planned to purchase 6000 basic systems for an estimated price of $29,276,679. Each basic system sold for $1,799 with an additional $300 for maintenance. The Air Force planned to use the personal computers in a stand-alone role but required that capabilities must exist to communicate with other DOD systems. Furthermore, all hardware and software must be compatible and transportable among the personal computers. Likewise, all hardware and software must not require a technician or special tools for installation. The users must be able to perform the installations using simple tools and following an installation manual. The contract was a twelve month fixed price contract with a twenty-four month extension option. The Air Force considered and planned for the expansion of each personal computer system as individual requirements dictated. In this respect, the contract specified the requirements for optional hardware. This hardware consisted of memory, floppy disk drives, hard disk drives, dot matrix and letter quality printers, modems, and plotters. These items also were purchased at a

fixed price as was the maintenance of all hardware items. Additional software, also available at a fixed price, included BASIC, COBOL, FORTRAN, and Pascal compilers, a word processor, a relational data base management system, a spreadsheet, and a graphics package.

Unfortunately for the Air Force, IBM entered the personal computer market in 1982. IBM used and supported the DOS operating system and eventually DOS replaced CP/M as the "standard" operating system. With fewer software programs available for the CP/M operating system, the Air Force issued a new contract for acquisition of a DOS based personal computer[13]. From the experience gained when implementing the 1983 personal computer contract, the Air Force saw the need for more than one basic configuration. This contract specified three basic hardware configurations ranging in price from $1103 to $1658. The smallest system consisted of one 80286 processor (16 bit), 512 kilobytes of memory, two parallel ports, one serial port, one graphics port, and two floppy disk drives. The intermediate system replaced one floppy disk drive with a twenty megabyte hard disk. The advanced system added an extra 640 kilobytes of memory to the intermediate system. Each hardware

---

[13] The following information is found in Contract No. F19630-86-D-0002, AFCAC Project 254, AFCAC, Hanscom Air Force Base, Maryland 01731, 28 February 1986.

configuration had the same basic software. This software included Microsoft DOS version 3.1, Microsoft BASIC 3 interpreter, Microsoft Windows, and diagnostic software. As with the other contract all basic and optional hardware and software were available at a fixed price. This contract lasts for twelve months with a twenty-four month extension option. The estimated price was $242,876,450.

The Air Force has performed an excellent job on standardizing data processing systems used Air Force wide. Some of these systems include accounting, finance, personnel, and supply. On the other hand, data processing systems not used Air Force wide lack standardization. These nonstandard systems may exist within a certain command or among a couple of commands. Because of this nonstandardization, communication between these systems may become impossible. Furthermore, a duplication of the design effort exists between these systems.

One of these nonstandardized systems is EWIR[14]. Because of security classifications, only an overview of this system is presented. Several locations[15] process the data gathered from many locations. The processing

---

[14] Some of the information presented in this section was obtained while working on the EWIR system at Electronic Security Command, San Antonio, Texas.

[15] These locations include Electronic Security Command, San Antonio, Texas; Foreign Technology Division, Dayton, Ohio; National Security Agency.

locations use hardware ranging from a Sperry 1184 to an IBM 360/370. In addition, the Army uses a CDC CYBER 160/170. Having these vastly different hardware configurations, magnetic tapes are the only means of communication between these systems. Similarly, the Sperry 1184 required development of special routines and procedures to process the tapes from the IBM 360/370.

Along the same vein, Syracuse Research Corporation (SRC), a government contractor, developed a subsystem for EWIR called Tools for EWIR Production (TEPRO). TEPRO consists of thirty-five features to aid in the analysis of EWIR data. First, TEPRO performs technical quality control. This quality control checks numerical EWIR data for validity. TEPRO also produces a tape of all new and updated data. Unfortunately, TEPRO exists on seven vastly different hardware configurations. In reviewing the actual code of TEPRO, it uses hardware dependencies of the CDC CYBER 160/170. Additionally, SRC did not enforce coding standards among the different modules. It took SRC several weeks to install the TEPRO subsystem at Electronic Security Command (ESC). Likewise, when new features are added to TEPRO, seven different software packages require updating, testing, and installing at the different locations. Because of the nonstandardization of hardware configurations, each location needs SRC to install these

updates. If the hardware was standardized, each command could install the updates without the aid of a contractor.

Since each of the processing locations performs the same function, the Air Force must utilize the centralized design approach for this system. Because Foreign Technology Division in Dayton, Ohio, is centrally located in respect at the other locations processing the data, it should become the design site. The ADS manager and the users' committee will reside at this location. This committee will include representatives from Electronic Security Command, National Security Agency, Syracuse Research Corporation, the Army, and several of the locations gathering the data.

Another data processing system the Air Force must standardize consists of the various wargaming systems. The Air Force has realized the need for standardizing these systems, but its efforts have fallen short of what is required. To determine the requirements of wargaming systems in the Air Force, the Air Force established the Wargaming Review Group (WRG)[16]. The WRG reviewed and recommended policies and procedures to oversee Air Force wargaming requirements validation, funding, participation

---

[16] The information provided in this section is found in a document from General Larry D. Welch, U. S. Air Force, Vice Chief of Staff, Office of the Air Force Chief of Staff, Washington, D. C., 22 March 1985.

in joint activities, and establishment and enforcement of Air Force doctrine and strategy. The Air Force Operations Office was selected to perform these activities. Additionally, the WRG established a wargaming technical center of expertise at Air University to aid in the cross feed of information and help to prevent any duplication of effort. Furthermore, Air University has the task of exploring areas to increase the sharing and consolidation of all wargaming systems.

The WRG reviewed several wargaming systems. The first of these systems, the McClintic Theater Model, provides excellent simulation of ground forces. The Army initially developed this model at the Army War College on a Wang computer system. Recently the model was converted to operate on a DEC VAX 11/780 at the Warrior Preparation Center. To provide simulation of air power, the Air Defense Simulation System was incorporated into the McClintic model. Problems in software compatibility occurred when ESC incorporated several smaller models to the McClintic model. Although ESC used a DEC VAX 11/780, the graphics software was different and the conversion of four small models took six weeks[17].

---

[17] This information was obtained while the author worked on the project at ESC.

Next, the BLUE FLAG system at the Tactical Air Warfare Center (TAWC) provides procedural training in a theater-specific environment for Tactical Air Command battle managers. This model coordinates the employment of fighters, reconnaissance, bombers, rescue, airlift, and refueling. Currently, this model executes on a combination of VAX and PDP 11 hardware. Fortunately, these two hardware configurations are compatible.

Air University currently uses the Command Readiness Exercise System (CRES). CRES is mainly an educational and research tool used to teach the thought processes necessary to plan, deploy, employ, and sustain forces in the battlefield. CRES prepares officers for operational and wartime command responsibilities. CRES along with the McClintic Model are becoming the major Air Force wargaming systems. Unfortunately, vast differences exist in their hardware configurations. As stated previously, the McClintic Model utilizes a VAX 11/780 while CRES executes on a CYBER 180/850 and 180/860. If the Air Force plans to utilize these two systems to their fullest extent, compatibility problems may not allow full interoperability between these systems.

To help achieve compatibility among DOD wargaming systems, the DOD has issued a contract obtaining standard hardware for all wargaming systems. Sadly, each wargaming

center is not required to use this contract or purchase a standardized hardware system. Furthermore, the Air Force Operations Office of AF/XOO is the office of primary responsibility for Air Force wargaming systems. Their responsibilities include determining manpower, facilities, and equipment requirements. Likewise, AF/XOO reviews all wargaming systems for adherence with Air Force doctrine. Also, AF/XOO was assigned the task to keep abreast of new technology and research. Unfortunately, AF/XOO does not have the properly trained personnel to perform this task. In addition, AF/XOO assumed total control over only two major wargaming systems. With two other major wargaming systems, the WRG recommended that AF/XOO only remain "cognizant" of these systems. Finally, TAWC was not included in the organization structure. The WRG recommended that TAWC just maintain close contact with Air University. Finally, several smaller wargaming systems were reviewed, but not included in the recommendations.

In the area of standards, the WRG did not recommend establishment of any basic hardware or software configurations. The WRG did encourage interoperability of hardware and software whenever possible. This interoperability has not been achieved as was evidenced in the previous descriptions of the wargaming systems.

Because of the varied wargaming functions each wargaming center performs, the Air Force should employ the decentralized design approach with centralized configuration management for managing wargaming systems. The ADS manager should reside at Air University and AF/XOO will only review each model for adherence to Air Force doctrine. The users' committee should include members from all Air Force locations using wargaming systems and not just the major wargaming systems reviewed earlier. Since wargaming involves the Army, Air Force, and Navy, a DOD users' committee must include members from all branches of the military. This committee must be located at the Pentagon and be responsible to the Joint Chiefs of Staff. This DOD committee will have absolute authority over each branch's wargaming systems. Currently, the Joint Chiefs of Staff have a contract for acquisition of standardized computer hardware for DOD wargaming systems[18]. Unfortunately, this contract is not being used by all locations acquiring wargaming systems. In fact, the Joint Electronic Warfare Center (JEWC), managed by the Joint Chiefs of Staff, recently purchased hardware for a wargaming system using a contract from McDonnell

---

[18] This is the Modern Aids to Planning Contract, MDA 903-85-D-0014, Defense Supply Service-Washington, Room 1D-245, The Pentagon, Washington, D. C. 20310-5220, 9 August 1985.

Douglas[19]. Furthermore, the JEWC and the Air Force Electronic Warfare Center (AFEWC) are physically located within 100 yards of each other, but these two organizations rarely coordinate their design and development efforts. To exemplify this point, these two organizations each have a valid requirement for digitized terrain data, but they will not store this data where it can be accessed by either organization. Each organization has acquired separate copies of the data and has duplicated the hardware needed to store and access this data. Since these two organizations both have a DEC VAX 11, centralized storage of this data is possible and desirable. In addition, the JEWC has requested contractor support in developing radar envelope and terrain masking routines. These are the same capabilities that the AFEWC delivered to the Warrior Preparation Center and could be easily transferred from the AFEWC to the JEWC.

Implementation at the Sites

Before design and analysis begins on the software, the user sites must select their staffs. Besides the site manager, this staff includes the configuration managers. With these people now in place, each site must obtain the

---

[19] This information was obtained by telephone from Captain Claude Baker, Joint Electronic Warfare Center, Electronic Security Command, Sna Antonio, Texas, August 1986.

approval of the ADS manager for all new requirements and purchases. Now analysis of the software can begin. For centrally developed systems, the ADS manager begins staffing the appropriate positions. Similarly for decentrally developed systems, the site manager performs the necessary staffing. The managers may acquire their staffs from sites currently developing software for each ADPS. This will consolidate and reduce the size of the staffs when using the central design approach. Upon completion of the analysis and design, a schedule for migration of the current systems to the new system can be determined, if appropriate. Additionally, at this time a request for proposal can be initiated for purchase of the necessary hardware. Next, the design staff initiates the conversion and testing of the current systems. The next chapter covers strategies for the conversion of current systems.

During the conversion of the software, the ADS manager determines the location for the operational test of the converted system. After the location is selected, the hardware configuration manager prepares the site for the operational test. This includes the determination of both the hardware and software configurations needed to fully test the software. Once the software passes the tests at the design site (more on testing later), the operational

site tests the software. Even though the software has passed the tests at the design site, the operational site should initially perform the same tests. This insures the same baseline at the test site as at the design site. For this operational test to be successful, all training materials must be completed and the staff at the test site adequately trained. Also, it will be helpful to have the development staff available at the test site during the operational test.

Upon completion of the operational tests, the design staff reviews the results and makes recommendations to solve any problems. Upon implementation of these recommendations, the operational test site performs a mini-test[20] (acceptance). When the system passes all tests, it is ready for transition to the sites.

In implementing the Phase IV Project, Gunter Air Force Station was designated to oversee the contractor and the design of the software. Furthermore, Gunter performed the configuration management of the project. The contractor developed and delivered the basic software functions. Each command identified their nonstandard software functions. Upon identification of these functions, each command assumed responsibility for converting and installing this

---

[20] The mini-test will be a scaled-down version of the operational test and will specifically address the areas where modifications have been made to the system.

software to the new hardware. The contract did not state a specific approach to use in performing this transition. The Air Force could have saved much time and energy by utilizing teams specializing in either Univac or Burroughs conversions. These teams could have been trained at Gunter and traveled to each command to aid in the conversion of the nonstandard software. In summation, the conversion of the nonstandard software could have been handled differently, but the centralized design of the Phase IV Project appears solid.

Similarly, both of the contracts for acquisition of personal computers were managed by a single organization, the Air Force Small Computer Office Automation Service Organization (AFSCOASO), and established a standard for acquisition, maintenance, and use of personal computers within the Air Force. Additionally, the Army, Navy, and Defense Logistics Agency could acquire personal computers using the 1986 contract. Furthermore, the Air Force, Navy, and Defense Logistics Agency were required to use only this contract when purchasing personal computers. Unfortunately, the Air Force did not specify what to do with the various nonstandard personal computer systems currently in use. AFSCOASO must gain control over all these various personal computer systems. First, AFSCOASO can centralize and standardize the maintenance contracts

for these systems. Using this strategy, only one maintenance contract will be necessary for each vendor. This will reduce the maintenance cost and also the amount of paperwork generated to oversee the contracts. Furthermore, when these nonstandard personal computer systems need replacing or expanding, the Air Force does not have a policy specifying that the standard personal computer system must be acquired. AFSCOASO must require each command to formulate plans for replacing its nonstandard personal computers with the standard personal computers. These plans will be implemented when the nonstandard systems no longer meet each command's requirements. This approach will let each command utilize their investments in the current systems while planning for the standardization of all personal computer systems.

In implementing the EWIR system, the users' committee will first analyze the process of gathering the data. This function may require different hardware and software than that of the locations processing the data. Since the data gathering process is less complicated than processing the data, the design and implementation of the data gathering system should be addressed first. Concurrently, the processing locations users' committees can analyze and review their requirements. This will allow the staff at the design site to apply the knowledge acquired on the data

gathering system to the system for processing the data. In addition, the ADS manager must include the TEPRO system in the analysis of the EWIR system. Currently, any location processing the EWIR data can request a change or update to the TEPRO system without the inputs or approval of the other processing locations. To exemplify this problem ESC has recently contracted with SRC for major updates to the TEPRO system without asking any of the other locations for their inputs. SRC implements any requested changes and then asks the other locations if the they want these changes; therefore, SRC has several implemented versions of the TEPRO system. Consequently, the EWIR ADS manager must also gain total control over the TEPRO system. With the ADS manager exercising control over all aspects of the EWIR system, the users' committee can carefully analyze the requirements of both the data gathering and processing locations.

Finally, the Air Force must apply the decentralized approach to the management of wargaming systems. The Air Force has done a good job of developing wargaming centers which perform certain functions. For example, the Blue Flag system provides tactical wargaming simulation. Likewise, the Warrior Preparation Center provides training for U.S. and European commanders using the entire European Theater and incorporating all NATO resources. The Air

Force wargaming locations along with the DOD wargaming locations have two basic hardware configurations. At the centers where actual battle simulation occurs, this hardware mostly consists of Digital Equipment Corporation PDP 11's and VAX 11's. Additionally, this is the hardware that the Joint Chiefs of Staff have acquired in the Modern Aids to Planning Contract. Since this hardware configuration is the de facto wargaming system and a DOD contract exists for purchasing this hardware, this contract must be utilized for all future wargaming systems. Unfortunately, the CRES system used at Air University executes on two CYBER 180's. Since this system is currently in the second phase of a three phase development effort, the development and implementation of this system should continue using the CYBER 180 system. Although using two vastly different hardware configurations in the same functional area does appear to violate the requirement for hardware standardization, the VAX and PDP systems are better suited to simulation and graphic displays and the CYBER system is best used in the role of determining and analyzing thought processes. Consequently, the hardware is designed to optimize and enhance the effectiveness of the software. This idea was stated earlier and should override most other considerations when determining the hardware that best suits the software. In summation, the DOD took a

major step toward standardizing requirements between the Air Force, Army, and Navy with the personal computer contract. Furthermore, contracting, maintenance, and training expenses will be substantially reduced by consolidation of requirements and volume purchasing of personal computers. In addition, the approach used with Phase IV Project has shown that standardized data processing systems can be implemented in the Air Force. Fortunately, the Air Force and DOD have identified the problem of incompatibility among wargaming systems. With incorporation of the ideas concerning management of wargaming systems, the Air Force may easily address and resolve this problem of incompatibility. Finally, the standardization of the EWIR system can show the Air Force the effectiveness of the centralized design approach on data processing systems not used Air Force wide.

CHAPTER V

CONVERSION OF CURRENT SYSTEMS

The Air Force has a large investment in software; therefore, it is not feasible or economical to discard this software. Consequently, the Air Force should consider converting the software to the appropriate hardware. Again, the importance of selecting the hardware to fit the software demands repeating. Because conversion of software does not enhance the software's capabilities, any organization should convert its software only if the converted software is to be enhanced at a later date. In other words, it is useless to convert software without subsequently enhancing that software. Any organization should avoid purchasing new hardware before the necessity to enhance the software exists. Once an organization determines their software needs enhancing, the organization must address and resolve several issues.

Planning

As with any major project, the organization must plan for the conversion. The planning phase breaks down into several smaller phases.

First, the organization enters the pre-conversion phase. In this phase, the organization analyzes the current workload [30, p. 67]. The organization appoints a small staff to perform this analysis. During this analysis, the staff classifies the current expenditures of the data processing function into hardware, software, supplies, and operations [26, p. 22]. Along this vein, the staff determines the expenditures and requirements of any future systems. Next, each system manager reviews his software. During this review, each manager purges his software. "Due to large workloads and backlogs, very little time is typically spent identifying and eliminating, on a regular basis, old or outdated software from the environment" [24, p. 20]. Once the outdated software has been eliminated, Wolberg suggests that the managers perform a software inventory and review. Each manager places the software into one of several categories (applications software, hardware dependent routines, and vendor supplied routines). The first category is applications software. For each application the manager lists the number of lines in each program along with all associated data files.

Additionally, the manager gathers any documentation and the usage statistics for each application and develops conversion packages for each program [30, p. 67]. Furthermore, the conversion package consists of the current and future memory and external storage requirements for each application. In the area of storage requirements, the package must contain the current and future sizes of all data files [26, p. 22]. Each data file description contains a record lay-out, record size, file usage, and file access method. Besides this information, the package covers the interfaces and interrelationships between programs and files. Moreover, privacy and security constraints are considered and resolved [24, p. 15]. Lastly, the conversion package contains a program and cross-reference listing for each application program. According to Wolberg, these two products will become the most important documents when the actual conversion begins [30, p. 15].

Next, each manager determines which programs cannot be converted. These programs may include assembly language, hardware dependent, or magnetic tape routines. If compatible routines do not exist on the new hardware, these routines will impact the conversion schedule dramatically. This impact results from the need to completely redesign, code, test, and finally implement these routines. This

redesign necessitates removing personnel from performing their normal duties. Additionally, if other programs require these routines, conversion of other programs must wait until completion of the new routines. The conversion of the new routines and associated programs now becomes a sequential process requiring additional time to complete the entire conversion.

Upon completion of the inventory of the application packages and nonconvertible routines, vendor supplied routines require analysis. These routines may entail reviewing the data base management system, utility programs, compilers, telecommunications software, and graphics packages. The manager should investigate options for handling these functions on the new hardware [30, p. 67]. With the growth of standardized personal computer software packages, these functions may be easily transported to the new hardware. Additionally, the vendor of each package may have the same package available for the new hardware. Once the manager determines the best solution for each package, Wolberg advocates that upper management reviews all recommendations to insure standardization. Finally, the manager examines and updates any programming standards [30, p. 67].

Upon completion of the software inventory, upper management determines if the conversion should continue. If management authorizes the conversion, planning can begin [30, p. 68]. Concurrently with the planning stage, a review group is formed. The Federal Conversion Support Center utilizes this group to perform a software inventory validation. The review group does not consist of the same people who performed the initial inventory. This group finds and resolves any inconsistencies in the inventory. The software inventory validation helps insure that no software is overlooked [24, p. 16]. Also, each manager with the aid of the users of each system develops test data for each program [24, p. 15]. Each program is then run using this test data. Quality control assumes responsibility for saving all files. This includes all files before and after the running of each program. These files are printed and a copy kept with the program listing. Quality control also maintains control over all test data. During a conversion at ESC, the programmers had access to the test data. When the output from a converted program did not match the stored output, the programmers changed the test data[21]. Consequently, quality control must maintain absolute control over the test data.

---

[21] This information was obtained during the conversion of an Air Force computer system.

Since conversion requires a firm commitment, upper management down to the software development staff must commit the necessary resources and time for completion of the project. During this phase, management identifies and begins training the conversion chiefs. The conversion plan resolves several questions. First, the plan identifies the objectives and methods of the conversion. This includes the organizational structure and responsibilities of the conversion staff. This may include the formation of conversion teams or the use of contractors [24, pp. 35-6]. The plan reviews where the organization is today and the organization's future goals [24, p. 5]. The conversion staff prioritizes each system and determines a tentative conversion schedule. If possible, the schedule includes plans for the conversion of the simple and small systems first. Then, the schedule can gradually progress to the more complex systems [24, p. 55]. Thus, the staff trains on small, independent systems and can apply this knowledge to the more critical systems where mistakes have greater impact on the plan. Next, the conversion staff determines the needed resources. These resources include staff and machine resources along with supplies, travel, and training expenses [24, p. 51]. For each resource, the plan identifies any and all constraints [24, p. 5]; for example, the computer may be available only certain hours during the

day. Next, quality control determines and publishes its functions and responsibilities [24, pp. 35-6]. The staff develops a programmer's notebook containing the differences between the old and new systems. Also, this notebook contains the descriptions and interfaces to any library or system routines. Oliver states that this notebook should also include any differences between the compilers of the old and new systems [25, p. 2]. Finally, the notebook clearly shows how to use and access any new function, utility, or compiler. Using the plan, schedule, allocated resources, programmer's notebook, and people with conversion experience, the staff initiates a medium-sized prototype project [24, pp. 35-6]. The federal government has found that keeping the conversion approach simple aids to the effectiveness of the project. "Thus, the probability of success for the conversion release is maximized by staying with as vanilla a conversion as possible" [24, p. 31]. Simply stated, this means add new functions, major logic changes, interface changes, or file structure changes only as needed [24, p. 31].

> This is done to avoid intermingling and thereby compounding any translation errors or effects with modification errors or effects. The success of the conversion will be closely related to the adequacy of the controls which are applied during the production stage [25, p. 2].

During a conversion at Electronic Security Command, major changes were incorporated into the file structures. This resulted in changes to many programs along with the creation of many new programs. Consequently, it is wise to avoid creating new programs during a conversion project. Once the prototype project is completed, the staff makes recommendations and changes to resolve problems they encountered. Finally, the plan considers the risks and impact upon the organization. If problems occur during the conversion, the organization needs contingency plans [24, pp. 35-6].

As with any major project, the expenditures associated with a software conversion project demand consideration. Pachaury has found that rarely does a conversion project consume less than twenty percent of the development costs [26, p. 21]. In 1977, the General Accounting Office reported that the federal government spent over $450 million to modify programs to work on hardware other than that originally designed for [25, p. 2]. "A survey of (computer) user budgets ..., has indicated that computer users spend around ten percent of their budget in conversion" [7, p. 205]. After analyzing many conversion projects, Wolberg has established a sound basis for time estimation. Refer to Table 2 for a breakdown of these estimates.

TABLE 2

PERCENTAGE OF TIME SPENT ON
CONVERSION TASKS

TASK	PERCENTAGE
Planning	5%
Data Preparation	10%
Conversion	25%
Testing	45%
Installation	15%

SOURCE: [30, p. 48]

Likewise, Wolberg categorizes costs by type. First, automatic costs encompass the expense to purchase or develop conversion tools. Manual costs cover any part of the conversion where someone must manually intervene. Fixed costs comprise areas like computer time, supplies, planning, and installation of hardware [30, p. 48]. The approach an organization chooses determines the amount of money expended for each of these costs.

Since conversion of software depends on the number of lines of code converted, several formulas exist to determine the manpower and duration of the conversion project. The formula for manpower is: Manpower = 7.14 * $lines^{0.47}$ [30, p. 35]. In the above equation, lines represents the number of lines of executable code in a single program. The manpower requirements for each program

are then summed to determine the number of person-months[22] for each system. The manager should not make the mistake of equating person-months with duration. The duration equation is: Duration = 4.1 * $lines^{0.22}$. Lines still represents the executable lines of code in each program. These values are again summed to produce the total duration for the conversion project. The duration is measured in calendar months [30, p. 35].

Oliver has observed that several other variables influence a conversion project. First, the old hardware impacts the conversion cost more than the new hardware. Second, a programmer does not need in-depth knowledge of a program to convert the program; however, if problems occur in converting a program, someone must have in-depth knowledge of the program. Next, complex programs will only cause scheduling problems if this complexity is not included in the time estimates. Furthermore, conversions involving a data base management system are not well documented or understood. In addition, models used in estimating costs and duration of new software systems prove useless [25, p. 9]. Upper management should understand that the conversion will shift many people from their normal duties and responsibilities. Additionally,

---

[22] A person-month is one person working for 152 hours on one project.

documentation may be missing and the programmer who wrote the program may have left the organization [25, p. 2]. Other factors affecting the conversion schedule include enforcement of standards used in the old system, degree of compatibility between hardware systems, the availability and quality of conversion tools, and support from the various vendors [26, p. 21]. Upper management must not impose a completion date on the conversion schedule. The amount of work should be the only factor influencing the completion date. During a conversion at Electronic Security Command, upper management did impose a completion date on the conversion staff. Using the days allocated to complete this project, each programmer had to completely convert two programs per week. This included testing and documenting each program as well as writing completely new job control streams. This project lasted four months longer than the projected ten months. With the planning phase completed, management must determine the appropriate strategy to employ during the conversion.

Strategies

"If there is a significant difference between the management of a conversion project and the management of a production project it is one of emphasis: a conversion project requires more discipline and stricter adherence to

procedures" [30, p. 62]. Wolberg states that control and management of a conversion project are the major sources of problems. Software conversion becomes an assembly line process with no ingenuity on the part of the staff required for completion of the project. Conversion involves more managerial control than technical expertise. Likewise, more management problems than technical problems occur. Many discrete tasks require management control for the conversion of each program. Some of these tasks include test data preparation, software inventory, parallel testing, and quality control. Similarly, software packages exist which perform many of the conversion steps [30, pp. 61-2]. In addition, a successful conversion project depends on previous conversion experience and adherence to established standards and controls [25, p. 5].

In the next step, management decides on the appropriate organizational strategy to employ for the project. According to Oliver, management must realize that a conversion project is a full-time job. This means temporarily removing all current responsibilities from the conversion staff. Thus, personnel not involved in the conversion must assume these additional responsibilities [25, p. 6]. Consequently, the users must understand that severe delays may occur to any request for assistance.

Next, management must decide whether to perform the conversion using in-house personnel or hiring contractors. Wolberg states that if the project is small and will last less than three months, the application staff responsible for each system may perform the conversion. Using this approach on projects lasting longer than three months will create morale problems [30, p. 69].

On the other hand, if the project will last longer than three months and in-house personnel are used, a good approach utilizes a team of conversion specialists. If the organization lacks experienced conversion personnel, the organization should hire experienced consultants to handle the management and control of the conversion. The team of conversion specialists move from system to system helping the application staffs convert their systems [30, p. 69]. Besides the team approach to software conversion, two other in-house approaches deserve consideration. First, the assembly line approach uses a different individual to perform each step of the conversion. These steps include developing the conversion package and the test data for each program. Likewise, different individuals write the job streams, convert the programs, write the documentation, and perform quality control. The assembly line approach requires less training since each individual assumes responsibility for only one stage of the conversion.

Furthermore, because the project can be broken into distinct steps, a time study may be performed during the prototype project. Large conversion projects should utilize the assembly line approach [24, pp. 46-7]. The Phillips Corporation used this approach when converting a system containing over 40,000 programs. These programs had over 40 million lines of code and the project lasted from June 1978 until December 1982. Phillips' strategy involved an assembly line approach. Each person performed a single function on each program. Although this approach is boring and monotonous to the conversion staff, this approach maximizes the efficiency of the staff. To aid the conversion staff, Phillips had a large investment in automated conversion tools [30, p. 78]. Figure 9 depicts a suitable organizational structure for a conversion project utilizing the assembly line approach. Wolberg uses the project coordination board as an interface to the users. The tool maintenance and support group develops and supports the software conversion tools. The conversion line leader supervises and controls the conversion of several subsystems. The technical coaches have conversion experience and provide assistance during the preparation and implementation phases. Finally, the trouble-shooters handle any unforeseen problems or troubles [30, pp. 83-4].

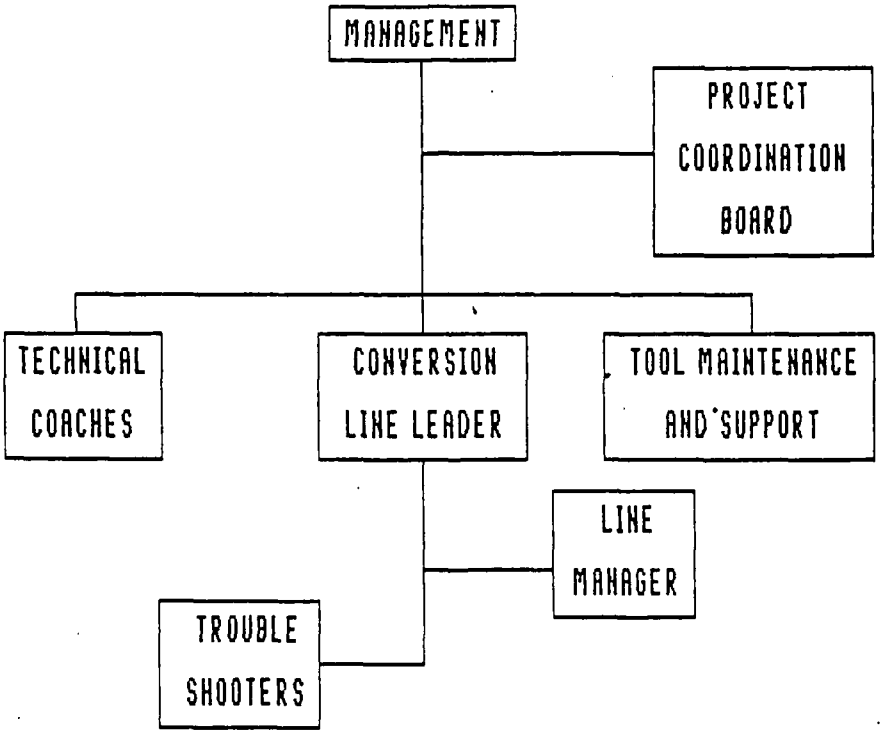

Figure 9. Organizational Structure for Software Conversion [3, p. 83]

Conversely, when a shortage of skilled personnel exists, management may consider using the job shop approach. In this approach, each individual performs all the steps of the conversion project except for quality control. The job shop approach necessitates a longer training and learning period since all individuals must acquire knowledge of all the conversion steps [24, pp. 46-7]. The job shop approach provides the most flexibility and all individuals become well versed in every step of the conversion [24, pp. 47-8].

On the other hand, the organization may choose to contract the entire conversion project. Consultants can supply the needed experience the organization lacks. First, contractors may furnish expertise during the conversion planning phase. In addition, the time and manpower estimates derived by a contractor may be more realistic and not as optimistic as the client's estimates [25, p. 2]. Furthermore, "conversion requires careful planning and it (conversion) should not be attempted without the services of professionals" [25, p. 9]. By employing contractors, many of the client's personnel remain free to perform their routine duties [30, p. 69]. Similarly, these in-house personnel may not be the best qualified to perform the conversion. "If anything, they (the in-house personnel) are the worst qualified since they

probably cannot resist the temptation to improve the system while converting it" [25, p. 7]. Unfortunately, even if the conversion is contracted, the client must utilize in-house personnel to prepare the statement of work, manage the contract, and oversee the conversion project [24, pp. 45-6]. In preparing the statement of work, the organization should allocate twelve to twenty-four weeks. The statement of work requires each bidder to prepare a technical response and a separate cost proposal. The client reviews each document separately. Over a four week period, one group reviews the technical responses and another the cost proposals. Each group ranks the proposals and submits the results to upper management who then selects the contractor and awards the contract. Preparing and awarding a contract for a conversion project may take between five and nine months[23]. If the project utilizes both contractors and in-house personnel converting the programs, it is imperative not to intermix these groups. The federal government has found that this intermixing of contractors and in-house personnel may cause conflicts and supervisory problems [24, pp. 47-8].

---

[23] This information on contracting was obtained from personal experience.

Since any conversion project requires strict adherence to procedures, Wolberg has observed that software conversion tools prove invaluable for projects greater than 5,000 lines of code. These tools reduce the amount of human intervention and therefore reduce the possibility of human error [30, p. 161]. "Experience with large conversion and improvement projects has shown they (the projects) cannot be successfully completed without the aid of automatic tools" [24, p. 64]. Although the price of these tools requires a large expenditure of money, management can justify this expense by the time saved eliminating many hours of manual work. In addition, as the size of a project increases so does the cost effectiveness of these tools increase [24, p. 64]. These tools relieve programmers from repetitive, boring tasks and uniformly enforce standards and changes [24, p. 65]. Because of the cost and unproven reliability of in-house development, the organization should purchase the necessary software tools. During the prototype phase, the conversion staff tests the effectiveness of the tools [24, p. 66]. The book, Datapro, Datamation, Computer Decisions, and the U.S. Federal Conversion Support Center have extensive lists of these tools, their prices, and vendors [30, p. 151]. The conversion staff can utilize these tools to perform many functions, including statement alignment and indention,

elimination of optional keywords, punctuation, placement of comments, and identification of potential problems [30, p. 90]. Furthermore, tools exist to create and load test data, convert the files between the old and new hardware, and compare test results of the two systems [25, p. 2]. Finally, software exists to automate many of the project management functions [30, p. 150].

Implementation

After conversion of each system, quality control must test and approve all the programs. Quality control becomes involved during the planning stage by ensuring the adequacy of the data used to test each system [25, p. 10]. According to Evans, several strategies exist for performing quality control. First, the organization may employ an independent test team approach, which gives the greatest degree of objectivity and independence. On the other hand, this approach presents many organizational and implementation problems along with being the most expensive approach for quality control. Second, quality control may exist within the data processing organization, but independent of software development [14, pp. 88-9]. In this approach, quality control may reside within the computer operations staff. This approach still allows independence and objectivity in addition to reducing the

organizational and implementation problems. The last approach integrates the test team with the software conversion team. Utilizing this approach, fewer software errors occur. Conversely, Evans finds that the quality control staff cannot effectively plan the testing phase since they must also perform the conversion of the software [14, pp. 88-9]. The second approach presents the most effective approach to quality control.

In a conversion project, quality control's main responsibility is testing the converted software. Since testing is a sequential process, quality control should not prepare detailed plans more than six weeks in advance [14, p. 51]. As stated in the beginning of this chapter, the time required to test the software depends on the amount of functional change incorporated into the converted software. These changes may introduce new errors to the programs. On the other hand, these changes may remove an undetected error from the old software. During the test phase, it may become impossible to determine which of these two cases may be true. Additionally, if major changes are introduced to the software, it may become impossible to compare the results of the old and new systems; therefore, conversion of the software must occur before any subsequent enhancement. Since software conversion differs from software development, a different strategy exists for

conversion testing. In software development testing, the strategy tests each program before integration into the complete system; however, conversion testing tests the entire system first [30, p. 82]. Wolberg states that testing of the individual programs only occurs if the system test does not produce the correct results. Phillips Corporation successfully used this testing approach into its large conversion project [30, p. 82].

Prior to the completion of testing, the conversion staff begins training the users. This training focuses on the differences between the old and new systems. The training includes familiarizing the users with any new hardware, procedures, and policies. The organization should have a full-time training staff consisting of trained teachers and not technical experts [24, p. 83]. The training staff develops plans for the training of the users. Each course plan includes specifying who performs the training, who is trained, the course content, when the training will occur, and what resources will be needed. These resources encompass the necessary facilities, computer hardware, computer time, study material, and teaching aids [24, p. 78]. To be effective, the training sessions integrate classroom teaching with hands-on experience. This training must occur just prior to the users' actually using the new system.

Once the new system has passed the quality control tests and the users have received the proper training, transition and acceptance of the new system begins. The transition plan must clearly specify who has responsibility for accepting the new system along with the criteria for this acceptance [24, pp. 191-92]. The transition plan ensures that the new system is operational before removal of the old system [24, p. 187]. When transition begins, quality control must ensure that the data files on the old and new systems are exactly identical [12, p. 205]. Several approaches exist for transition between two systems. Factors like available manpower and computer time have a major impact on the transition approach [24, p. 192]. First, the conversion staff may decide on an immediate transition approach. Unfortunately, if problems with the new system occur, the users cannot return to the old system. Although this approach is the least expensive, it is also the riskiest, requiring very thorough testing of the new system. This testing includes the applications software, systems software, and the hardware. Although this approach has the shortest duration, user involvement remains moderate [24, pp. 196-97]. If the converted system performs a small, noncritical function, immediate transition may accomplish the desired results [22, p. 129]. On the other hand, the system may require a complete

parallel transition approach. In this approach, the users exercise the new system while simultaneously maintaining the old system. Any data input into the old system must be input into the new system. The users perform this testing for a specified period of time. When appropriate, the users compare the outputs of the two systems. When the system meets the acceptance criteria, the conversion staff removes the old system and the users take control of the new system. This approach has little or no risk since the old system is always available. Unfortunately, this approach requires more personnel than immediate transition. Furthermore, maintaining two computer systems necessitates additional financial expenditures. Since the users must duplicate their work on the new system, the users may need to work twice as hard as normal. Fortunately, this approach does provide an excellent operational test of the new system before the users accept the system [24, pp. 193-96].

After the users accept the new system, the conversion staff slowly phases-out direct support of the new system. According to Donaldson, management disbands the conversion staff and each software manager resumes control over the new system [12, p. 210]. During this several week phasing-out period, the conversion staff acquaints the maintenance personnel with the new system. Prior to

disbanding, the conversion staff performs a project evaluation. This includes a review of the successes and problems of the conversion. Finally, the conversion staff makes recommendations to incorporate into future conversion projects [22, pp. 130-31].

Enhancement

As stated in the beginning of this chapter, conversion of software does not increase the efficiency or effectiveness of the software. The software conversion project was only the prerequisite for future software enhancement [24, p. 27]. The individual software managers begin enhancing their systems on a program by program basis [30, p. 71]. This program enhancement focuses on identifying and eliminating inefficiencies in the software and maximizing the performance of the new hardware. If a program meets the functional requirements of the user but has problems in efficiency or maintainability, this program requires enhancement [24, p. 27]. Each manager reviews and analyzes their systems on a macroscopic and microscopic level. During the macroscopic analysis, each manager identifies programs which need enhancing--those that are not within a factor of two of their specified requirements. Wolberg suggests that the manager replace or reprogram these inefficient programs [30, p. 106]. This

reprogramming focuses on the input/output routines since these usually form the bottleneck in a system. Sometimes reprogramming does not require changing the programs. Wolberg has found that increasing the efficiency of the input/output routines may just necessitate increasing memory buffers, changing the job streams, blocking the files properly, or separating the files among different disks. Additionally, periodic reorganization of dynamic files increases the speed of data accesses [30, p. 126]. Next, sorting techniques may decrease the efficiency of the system. The managers should not assume that the sorting techniques on the old system work efficiently on the new system. Each manager may increase sorting efficiency by acquiring a commercial sorting package which optimizes the hardware environment. Furthermore, converting sorts from external storage devices to internal memory may dramatically increase efficiency [30, p. 131]. Lastly, each manager reviews the compiler and run-time options specified for each program. For example, a production program does not need the debug options activated. Wolberg has observed that removing these unnecessary options requires no program changes and is a quick and inexpensive method to improve program efficiency [30, p. 113].

Unfortunately, these methods may not increase system efficiency enough and it may be necessary to change some programs. The manager and programmers now perform a microscopic analysis of the actual code. The manager has several options at this point. First, if a vendor package is available, the manager can use this package to replace the program [24, p. 23]. Next, the programmer reviews the file access methods and where the opening and closing of files occurs [30, p. 126]. In this approach, the programmer removes and reprograms the inefficient code in a system. Compared to completely redesigning the system, reprogramming involves about fifty percent of the effort to redesign the system [30, pp. 43-7]. Finally, if none of the previous steps improve the efficiency of the system, the system is a candidate for redesign. Unfortunately, redesigning a system utilizes the greatest amount of time, money, and resources; therefore, a manager should avoid redesigning a system whenever possible. Fortunately, if good documentation exists for the system and some of the staff who developed the system remain, Brown states that this redesign should consume only twenty percent of the

original development expenditures [7, p. 314]. As with the conversion project, each manager performs a prototype enhancement project and uses this project to gain knowledge and experience. The manager incorporates this information into the enhancement plans and strategies [30, p. 71].

CHAPTER VI

CONFIGURATION MANAGEMENT

Configuration management encompasses change control[24], configuration item identification[25], and product approval [14, pp. 42-3]. As part of a directive on the subject, the DOD defines configuration management as

> A discipline applying technical and administrative direction and surveillance to (1) identify and document the functional and physical characteristics of a configuration item; (2) control changes to those characteristics; and (3) record and report change processing and implementation status [9, p. 21].

Although this directive exists, the Air Force, Navy, and Army implement configuration management differently [22, p. 136]. The traditional view of data processing configuration management relates to the consistent labeling, tracking, and change control of the hardware elements of a system. Bersoff states that this view treats each piece of hardware as an individual item while treating all the software as a single entity [3, p. 6].

---

[24] Change control is the process of identifying and tracking the modifications made to a configuration.

[25] Configuration item identification is the process of determining the items (software routines, hardware) which will be controlled and tracked as a single entity.

> This (treating software as a single entity) is a consequence of the observation that the software developer creates a product only once, and then modifies it as required to meet changing needs. This attitude is reinforced by the traditional mode of operation, in which the computing capability (software) is developed and used at a single site [19, p. 20].

With centralization of software development, configuration management plays a very important function. "No matter where you are in the system life cycle, the system will change, and the desire to change it will persist throughout the life cycle" [4, p. 335].

Plan

The configuration management plan specifies the level of configuration management. This level identifies what items will be managed under the plan. These are called configuration items [11, p. 27]. Additionally, in the DOD, the plan defines and outlines the responsibilities between the government and the contractor [11, p. 21]. The plan specifies a standard procedure to control, manage, and maintain the level and number of changes to a system [24, p. 75].

Baseline

The configuration manager must establish a starting point before changes to a system can be controlled and managed. The baseline is this starting point. The

baseline encompasses all the resources and configuration items needed to meet the current requirements. Bersoff has identified five baselines the manager must maintain for each system. The functional baseline covers the system concept [4, p. 47]. The hardware and software have no allocated functions at this point. The allocated baseline establishes the system baseline and divides the functions to either hardware or software. Next, the design baseline organizes the hardware and software requirements. Completion of these requirements occurs during the product baseline. From the product baseline, management generates hardware and software proposals. The operational baseline consists of the system currently used [4, p. 51]. After a baseline is updated, the manager examines and reviews the integrity of the system. This review focuses on the satisfaction of and adherence to the requirements [3, p. 8].

Change Control

To maintain and control these baselines, the configuration manager must establish and enforce procedures to manage the changes to any of the baselines. "Change has no meaning if it cannot be described with respect to a fixed reference point" [4, p. 100]. The fixed reference points are the current baselines. The configuration

manager establishes and chairs the configuration control board. Anyone can request a change to a baseline, but the configuration control board reviews and approves all changes [22, p. 139]. Metzger classifies these changes into two categories. If the change does not impact any of the current baselines or plans and requires little or no expenditure of funds, the board implements this change immediately. On the other hand, if the change forces an alteration of the baselines or plans, the board must carefully incorporate this change into the baselines [22, pp. 88-9].

The organization establishes two subordinate configuration control boards. One board supervises the hardware changes while the other manages the software changes. Both boards consist of the project manager, systems engineer, quality control manager, and user representative. The hardware configuration control board includes the software configuration manager. At the same time, the hardware configuration manager sits on the software configuration control board [4, p. 185]. Martin Marietta requires a change request plus an impact summary before the configuration control board will review any changes. The board uses these documents to approve or disapprove each request [19, p. 146]. Boeing Corporation allows changes to the operational baseline only every four

months. Additionally, Boeing's approach to configuration management allows the board to approve the changes, allocate the resources, determine the impact of the request, and prioritize the approved changes [19, p. 143].

As stated in the beginning of this chapter, the DOD uses configuration management, but each service implements it differently. The Air Force along with the DOD must standardize their configuration management procedures in order to allow consistent control over DOD projects. Consequently, any branch of the military can perform the configuration management of any given product. First, each ADS manager must establish the configuration control board and the two subordinate boards. Martin Marietta's policy of including an impact summary with any change request will aid in eliminating nuisance or petty changes to a system. Additionally, allowing changes to the operational baseline once every four months will lessen the amount of paperwork and documentation changes made to a given system. Furthermore, this will allow tighter control over the system. On the other hand, the DOD should not allow the configuration control board to approve change requests, allocate resources, or assign priorities. Their only function is to maintain the current baselines, determine the impact of change requests on the current baselines, and recommend approval or disapproval of these change

requests. The ADS manager will approve the requests and allocate the resources.

Hardware

Hardware configuration management tracks the utilization of each component. These components include the processor, memory, disk storage, display devices, and any other hardware necessary to satisfy the requirements of that particular site. Fried says that the hardware configuration manager ascertains which hardware component is the limiting factor of each system and determines the cost effectiveness of replacing this component [17, p. 261]. If the manager does not replace that component, he checks the utilization of that limiting component each time the hardware or software operational baseline is changed. The hardware configuration manager analyzes the hardware utilization on each shift and replaces components with unacceptable downtime [17, p. 261]. Furthermore, the hardware configuration manager uses the functional and allocated baselines to determine the future requirements of the hardware. These future requirements range from the computer hardware to the facilities needed to house the hardware [17, pp. 231-32].

Software

> Software configuration management, like (hardware) configuration management, is defined as the discipline of identifying the configuration of a system at discrete points in time for purposes of systematically controlling changes to this configuration and maintaining the integrity and traceability of this configuration throughout the system life cycle. [4, p. 20].

Bersoff states that software configuration management involves four major functions. First, the software configuration manager identifies the software items to be controlled [3, p. 9]. Identifying these items is the most crucial step in imposing software configuration management [4, p. 100]. In identifying these items, the manager must determine at which level--module, unit, subsystem, or system--configuration management for each system will be applied. Next, the manager establishes and enforces procedures to control changes to these software items. Status accounting analyzes the effect any software baseline change will have on the overall system performance. Finally, auditing reviews the changes to the baselines and verifies that the specifications are satisfied [3, p. 9].

Software configuration management presents more problems than hardware configuration management. First, software is easier to change than hardware [3, p. 6]. Additionally, software changes may not be as physically noticeable as hardware changes. Furthermore, four

versions of each software configuration item may exist simultaneously. These different versions may exist in the functional, design, product, and operational baselines [3, p. 9]. In addition, the software configuration manager must maintain and update several documents associated with each software configuration item. These documents consist of the user's manual, operator's manual, design specifications, test material, source code, and executable code [11, pp. 28-31]. Keeping these documents consistent requires strict control [3, p. 7] and once software configuration management begins, any change to a software configuration item requires proper documentation and approval [28, p. 344]. On the Trident submarine project, CSC did not strictly enforce software configuration management procedures. As a result, CSC had problems identifying and controlling the current software operational baseline. This resulted in higher development expenses. On the other hand, CSC did strictly enforce software configuration management on the AEGIS and DD-963 projects. This resulted in effectively controlling changes to the operational baseline [19, pp. 185-86].
Configuration management, especially software configuration management, represents a major function in developing and adequately maintaining a data processing system.

CHAPTER VII

REDUCING SOFTWARE COSTS IN THE NEAR FUTURE

The ideas and approaches expressed thus far may not produce tangible results for several years; however, designing software to be portable may provide a solution in the near future.

> Too much time is currently spent in conversion due to hardware infatuation. Management has been stamped into changing hardware and software far too frequently. It should be possible for a user to choose a computer which meets his requirements for the next seven to nine years in terms of expandability, buy it and stay with it; but the average life of a particular computer in an installation appears to be three years [7, p. 205].

Portable Software

Portable software is software easily implemented on different hardware systems. Portability entails designing and developing strategies to assure hardware independence of the software. Brown has determined that several factors influence this degree of hardware independence. First, the nature of the application dominates this independence. Furthermore, the degree of hardware independence impacts

the portability of the software. Next, the size, structure, and complexity of the different hardware systems for which the software is designed dictates the amount of portability. Finally, developing portable software requires highly qualified and trained programmers, very rigidly enforced standards, and solid organizational policies and procedures [7, pp. 313-14].

Unfortunately, hardware dependencies may still exist in the portable software.

> Druseikis (1975) advocates separation structuring as a method of enhancing portability. This term (separation structuring) refers to the logical and physical isolation of machine dependencies [13, pp. 119-20].

In determining these hardware dependencies, Wallis suggests that the design staff analyzes all potential hardware machines for common features. The staff then incorporates these features into the design of the system [29, p. 49]. The design staff identifies the hardware dependencies early in the software design phase and then confines these dependencies to a few modules. During system testing, these machine dependent modules receive more thorough testing [29, p. 2]. In determining the amount of hardware independence, the design staff must realize that the cost of the software increases proportionally to the amount of

hardware independence [23, p. 123]. Consequently, the removal of some hardware dependencies will dramatically increase the price of the software.

> Despite the clear advantage of portability, the use of techniques in developing portable software is far from universal. Part of the reason for this is economic--portable programming requires extra care and professionalism so it is more expensive to write a program in a portable rather than a non portable, way [29, p. 5].

In addition, portable programs tend to require more memory than the equivalent non-portable programs [7, p. 125]. Furthermore, Wallis suggests that only the users want portable software. Having portable software, the users can change either the software or hardware. The users do not limit their selection to only a single vendor; therefore, the users can shop and save. On the other hand, hardware manufacturers do not encourage the use of portable software. Customers using portable software could easily change to a different manufacturer's hardware [29, p. 6]. Portable software exists in the personal computer market and customers seem more concerned with the prices than with the brand name of the hardware. Finally, software companies do not want portable software. These companies would lose revenues acquired from the conversion of software to different hardware [29, p. 6].

Fortunately, several advantages do exist for the use of portable software. First, the software expenditures can be spread among many hardware systems. The software written for the personal computer market clearly represents this situation. Second, Wallis also states that developing portable software requires a higher level of design and documentation. Thus, the design staff expends more time performing analysis and examining alternatives. Finally, designers, programmers, and users become portable. Once trained, these people can move to a different installation and not require additional training [29, pp. 2-3].

Design

In designing portable software, the staff must remain cognizant of many potential problems. First, the staff should avoid the use of character packing. Second, since word sizes vary from machine to machine, problems concerning numerical accuracy may arise. Next, the staff should not use operating system functions when designing the software [29, p. 48]. Besides not performing mixed-mode arithmetic, the programmers must fully parenthesize all numeric expressions [24, p. 117]. Although standards do exist for each language, these standards allow each vendor to include optional features. For example, the COBOL-74 standard has 300 reserved words.

Although this is the standard, IBM added forty-three new reserved words to its DOS/OS COBOL and excluded sixty-four reserved words required by the standard. Univac in its OS/4 COBOL added 11 new words while excluding 102 required words [30, p. 7]. Consequently, Dahlstrand advocates that the design staff identify a standard subset of each language and strictly enforces the use of only that subset [8, p. 22]. Fortunately, the DOD has identified the problems associated with allowing optional features to a programming language and forbids any optional features in the Ada language [29, p. 74].

To aid the design staff in developing portable software, several procedures may prove helpful. First, according to Dahlstrand, all the software must exist in normal form. Normal form means all software including subroutine libraries are available in source form. In addition, all files must exist in character, formatted form [8, p. 88]. Also, the use of common subroutine libraries increases the standardization and readability of the software. Furthermore, in most business applications, the use of libraries may reduce the amount of code between forty and sixty percent. Since the use of libraries entails less code, the software requires less maintenance [24, pp. 154-55]. Consequently, the modular design of the library routines forces modularity into the software

design. Finally, the design staff may acquire software to insure and enforce these standards. Filters are available which impose a subset of any language onto a software system [7, p. 37].

In implementing portable programming, the Air Force should utilize the ADS managers as the individuals responsible for incorporating this concept into designing new software. First, since each ADS manager already knows the different hardware systems in his ADPS, each manager can design his software to fit those hardware configurations. Additionally, the Ada language should be used whenever possible. On the other hand, if the use of Ada is not possible or feasible, each ADS manager should identify a subset of each language that will perform identically on all hardware configurations within the ADPS. The ADS managers must strictly enforce the use of these language subsets. Finally, each design staff must develop a common set of routines in areas such as input/output, numeric calculations, graphics, and data communications.

Implementation

Wallis states that testing of portable software involves two phases. First, the design site must exhaustively test the system. Second, Wallis advocates

testing each hardware dependent routine on the appropriate hardware [29, pp. 23-4]. According to Brown, this process of developing and implementing portable software requires a designer and installer. The designer develops and documents the software. Also, the designer has knowledge of the targeted hardware along with prior experience as an installer. To aid in a smooth implementation of the software, Brown suggests having the designer install the software at the first target site. From this experience, the designer can update the installation documentation if needed [7, p. 117]. This documentation will include the parameters for each hardware system and the software modules needed for each hardware system. In addition, this documentation contains the test data and the desired results. Finally, Wallis requires that the designer provide extensive documentation on any non-portable routines. The installer then assumes responsibility for developing these routines [29, p. 21]. Unfortunately, Brown finds that the installer has less experience and ability than the designer. Additionally, the installer lacks intimate knowledge of the software and the source hardware. Fortunately, the installer does have thorough knowledge of the target hardware [7, p. 118].

Because communication problems may arise between the designer and installer, Brown requires that the designer provide a glossary of terms to the installer. Likewise, the designer should assume that the installer is ignorant and not make any assumptions about the knowledge of the installer [7, p. 122]. Unfortunately, the design staff cannot eliminate these communication problems by using a designer as the installer. First, the designer lacks knowledge of the target hardware. Furthermore, the designer may have implicit knowledge concerning the installation procedure which is not stated in the documentation [7, p. 119].

Before the installation of the software at the target sites, each ADS manager must train individuals to install the software at these sites. During the design of the software, the staff develops the library routines first. Furthermore, the staff analyzes and documents the hardware dependent routines for development by the installers. After the library routines pass the tests at the design site, they are then tested at each target site. This will let the installers become familiar with part of the system before the actual installation. Additionally, the installers can develop and test the hardware dependent routines while the design staff completes the rest of the software.

New Trends

Although the following ideas for developing portable software have not been widely utilized, they do deserve further investigation. A study completed in 1985, emphasized removing software development from the critical path of computer systems development. This concept focuses on a software-first machine. This machine consists of hardware which is capable of simulating other hardware systems through the use of microcode. Using this approach, the design staff uses the microcoded hardware to design the software. During the system test phase, the staff acquires the actual hardware. Consequently, the hardware is now on the critical path and not the software. Additionally, software development begins before hardware acquisition. An Air Force organization has successfully utilized this approach. This organization acquired microcoded hardware which simulated the current hardware. One of the processors was microcoded to simulate the requirements of the new hardware. After the software was completed, the organization changed all the processors' microcode to simulate the new hardware [5, pp. 55-6]. Using this approach, no new hardware was needed to run the new

software. Unfortunately, not all hardware contains replaceable microcode. Consequently, this approach may be effective, but not usable in all systems.

CHAPTER VIII

SUMMATION

During the past fifteen years, several interesting trends have occurred in the data processing industry. First, the price of hardware has decreased between thirty and ninety-nine percent. On the other hand, software expenses have increased between 25 and 1100 percent. Likewise, salaries paid to personnel in the data processing industry have also increased between 30 and 335 percent. With this in mind, by implementing the author's recommendations of centralization of management, acquisition of hardware suited to the software function, and standardization of system design and configuration management the Air Force can take advantage of the decreasing hardware prices and offset the increasing software and personnel expenditures. These recommendations will place the authority and the responsibility associated

with each function at the appropriate level within the organization. Additionally, users will be able to share software and reduce their training expenses associated with both the data processing and users' organizations.

Any new implementation of procedures must resolve a variety of problems. First, incorrect or inappropriate management structures similar to those depicted in Figures 2 and 3 must be eliminated. Next, organizations must eliminate the "we can have it for you tomorrow syndrome" caused by inadequate planning and vague objectives. Additionally, the lack of software configuration management and the non-portability of software need to be resolved. Finally, remedying the inadequacy of decision-making structures must be considered. The Air Force has the basic decision-making structure in place, but does not use it to its fullest capacity.

The Air Force's decision-making structure is presumed during implementation of the author's recommendations. By incorporating the organizational structures shown in Figures 4, 5, and 6 when developing and maintaining data processing systems the Air Force will resolve the problems of inappropriate structures, responsibilities without authority, and standardization of configuration management. Next, all current data processing systems must be defined and placed into functional categories. For each

functional category an ADS manager will be assigned. With the ADS managers in place they, in turn, will select the organizational structure appropriate for their data processing systems and begin staffing their organizations as requirements increase. In other words, the ADS manager will only acquire additional personnel as the system progresses through analysis, design, development, programming, testing, and implementation. With the help of the users' committee, the ADS manager for each system must develop, maintain, and follow a set of long-range plans. These plans should address, but not be limited to the following items:

1. Where is this system heading?
2. Will this system interface with other systems in the future?
3. Will this system be implemented at any new locations?
4. How can new technology be utilized in this system?

Next, the decision must be made whether to convert and subsequently enhance the current system or completely redesign the entire system. Redesigning the system should be attempted as a last resort since this approach is more expensive and more risky than the convert and enhance approach. Furthermore, redesigning a system normally incurs a longer development time than any other design

approach. If redesigning the software is attempted, a phased implementation approach may provide the best results. Finally, when the system has successfully completed all quality control tests, the software is ready for implementation in the field.

Unfortunately, the task of completing the steps necessary for the implementation of the recommendations presented in this thesis is not without problems. The following are some of the problems any organization must overcome:

1. Resistance to change,
2. Determining the number functional categories,
3. Placing the systems into these categories,
4. Acquiring and keeping knowledgeable staffs,
5. Satisfying all users,
6. Overcoming long development times, and
7. Acquiring the funds.

On the other hand, many advantages are incurred by following the author's recommendations.

1. Fewer but larger systems mean increased competition for contracts leading to lower costs,
2. Fewer data processing personnel because of centralization,
3. Increased transportability of software,
4. Increased portability of both users and data processing staff,
5. Lower training expenditures, and
6. Fewer contracts to manage.

By slowly and carefully implementing the author's recommendations, the Air Force can take the lead in trimming data processing expenditures and set the standard for the other services to follow. In addition, if all branches of the DOD use a standard methodology in developing, managing, and controlling data processing systems then satisfying similar requirements between the different branches of the DOD will require only a single design effort. This standardization of DOD data processing systems is the ultimate goal of the author's effort in this thesis.

SELECTED BIBLIOGRAPHY

1. Baker, Terry F. and Mills, Harlan D. "Chief Programmer Teams." Datamation December 1973: 58-61.

2. Bentley, Colin. Computer Project Management. London: Heyden and Son, Ltd., 1982.

3. Bersoff, Edward H.; Henderson, Vilas D., and Siegel, Stan G. "Software Configuration Management: A Tutorial. " Computer January 1979: 6-13.

4. ----------. Software Configuration Management--An Investment in Product Integrity. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1980.

5. Boehm, Barry W. "Software and its Impact: A Quantitative Assessment." Datamation May 1973: 48-59.

6. Brooks, Frederick. "The Mythical Man-Month." Datamation December 1974: 45-52.

7. Brown, P. J. ed. Software Portability an Advanced Course. Cambridge: Cambridge University Press, 1977.

8. Dahlstrand, Ingemar. Software Portability and Standards. Chichester: Ellis Horwood, Ltd., 1984.

9. Dean, W. A. "Why Worry About Configuration Management." Defense Systems Management Review Summer 1979: 21-29.

10. Department of the Air Force. Automated Data Systems/Automated Data Processing Systems List. Design Center, Gunter Air Force Station, Alabama, 24 September 1983.

11. ----------. Contract No. F19630-81-D-0002 Base Level Data Automation Program Phase IV Configuration Management Plan (Implementation Period). Air Force Automated Systems Project Office(AFCC), Directorate of Contracting/PGZ, Gunter Air Force Station, Alabama, 2 September 1982.

12. Donaldson, Hamish. A Guide to the Succcesful Management of Computer Projects. New York: John Wiley and Sons, Inc., 1978.

13. Eckols, Steve. How to Design and Develop Business Systems--A Practical Approach to Analysis, Design and Implementation. Fresno, California: Mike Murach and Associates, Inc., 1983.

14. Evans, Michael W. Productive Software Test Management. New York: John Wiley and Sons, Inc., 1984.

15. Foss, W. B. "Seeking System Productivity? User--DP Interation Critical to Design Process." Computerworld 26 April 1982: 2.

16. Frankenhuis, Jean Pierre. "How to Get a Good Consultant." Harvard Business Review November-December 1977: 133-139.

17. Fried, Louis. Practical Data Processing Management. Reston, Virginia: Reston Publishing Company, Inc., 1979.

18. Fuchs, Jerome H. "Management Consulting Services Reduce Costs." The Office January 1979: 130-132.

19. Glass, Robert L. Modern Programming Practices--A Report From Industry. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1982.

20. Kennedy, James H. "Management Consultants and Conflicts of Interest." Dun's Review March 1978: 117-121.

21. Lehman, John H. "How Software Projects are Really Managed." Datamation January 1979: 119-129.

22. Metzger, Philip W. Managing a Programming Project. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1973.

23. Muxworthy, D. T. ed. Programming for Software Sharing. Dordrecht, Holland: D. Reidel Publishing Company, 1983.

24. Office of Software Development. The Software Improvement Process--Its Phases and Tasks (Part 1 of 2). Federal Conversion Support Center, Washington D. C., July 1983.

25. Oliver, Paul. "Software Conversion and Benchmarking." Software World 16 February 1979: 2-11.

26. Pachaury, Vijaykar. "The Throes of Change." Data Processing May 1977: 21-23.

27. Peters, Lawrence. "Managing the Transition to Structured Programming." Datamation May 1975: 88-96.

28. Tausworthe, Robert C. Standardized Development of Computer Software. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1977.

29. Wallis, Peter J. L. Portable Programming. New York: John Wiley and Sons, Inc., 1982.

30. Wolberg, John R. Conversion of Computer Software. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1983.

ABBREVIATIONS

ADPS: Automated Data Processing System

ADS: Automated Data System

AFCAC: Air Force Computer Acquisition Center

AFEWC: Air Force Electronic Warfare Center

AFSCOASO: Air Force Small Computer Office Automation Service Organization

CRES: Command Readiness Exercise System

CSC: Computer Sciences Corporation

DEC: Digital Equipment Corporation

DOD: Department of Defense

ESC: Electronic Security Command

EWIR: Electronic Warfare Integrated Reprogramming

GNP: Gross National Product

JEWC: Joint Electronic Warfare Center

NASA: National Aeronautics and Space Administration

NATO: North Atlantic Treaty Organization

SDC: Systems Development Corporation

SRC: Syracuse Research Corporation

TAWC: Tactical Air Warfare Center

TERPO: Tools For EWIR Production

WRG: Wargaming Review Group

VITA

William Joseph Fetech w[redacted].

[redacted]eo William Fetech and the former Margaret [redacted]. He was the third of four children and is married to the former Debora K. Siemering of Kyle, Texas.

After his father retired from the Air Force, his family settled in New Braunfels, Texas, where he graduated from New Braunfels High School in 1977. In 1982, he graduated magna cum laude from Southwest Texas State University, San Marcos, Texas, with a B.S. in Computer Science and a minor in Mathematics. Upon graduation from college, he accepted his commission as a second lieutenant in the U. S. Air Force and received the Distinguished Graduate Award from the Air Force.

In his first assignment, he served as Electronic Warfare Systems Analyst, Electronic Security Command, San Antonio, Texas. He accepted the assignment to Trinity University in 1985. Upon graduation, Captain Fetech will serve as Chief Systems Software Support, Randolph Air Force Base, San Antonio, Texas.